Kullanımı
Baslangıç Egitimi

Getting Started with Digital Identifiers and Credentials for Enterprise Architects & Developers

Michael Herman

Self-Sovereign Blockchain Futurist, Architect, and Developer
Hyperonomy Digital Identity Lab
Parallelspace Corporation

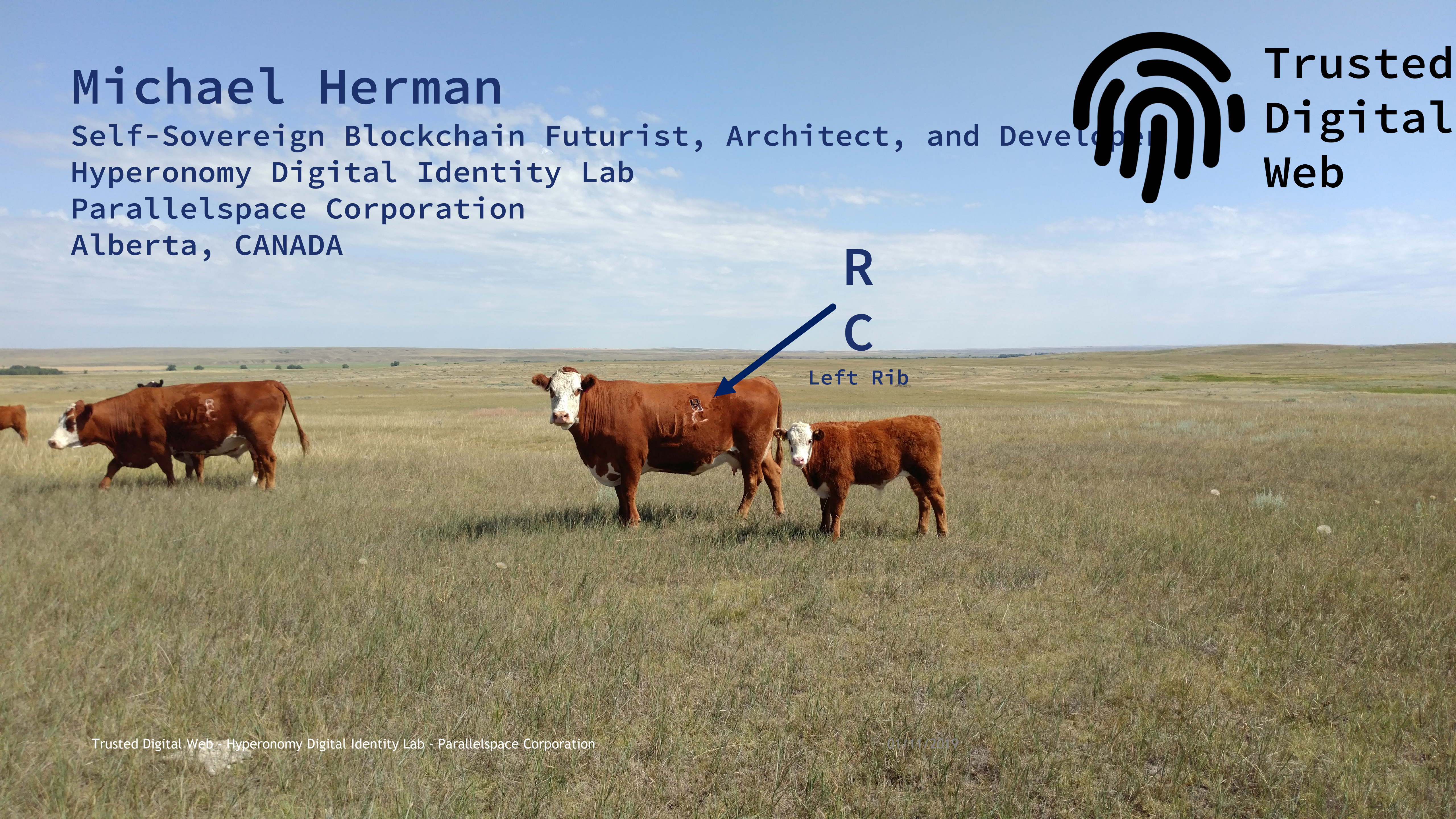
ALBERTA, Canada

#indygsgea   UKTAS/uktas2019   belgem.io

Michael Herman

Self-Sovereign Blockchain Futurist, Architect, and Developer
Hyperonomy Digital Identity Lab
Parallelspace Corporation
Alberta, CANADA

Trusted
Digital
Web

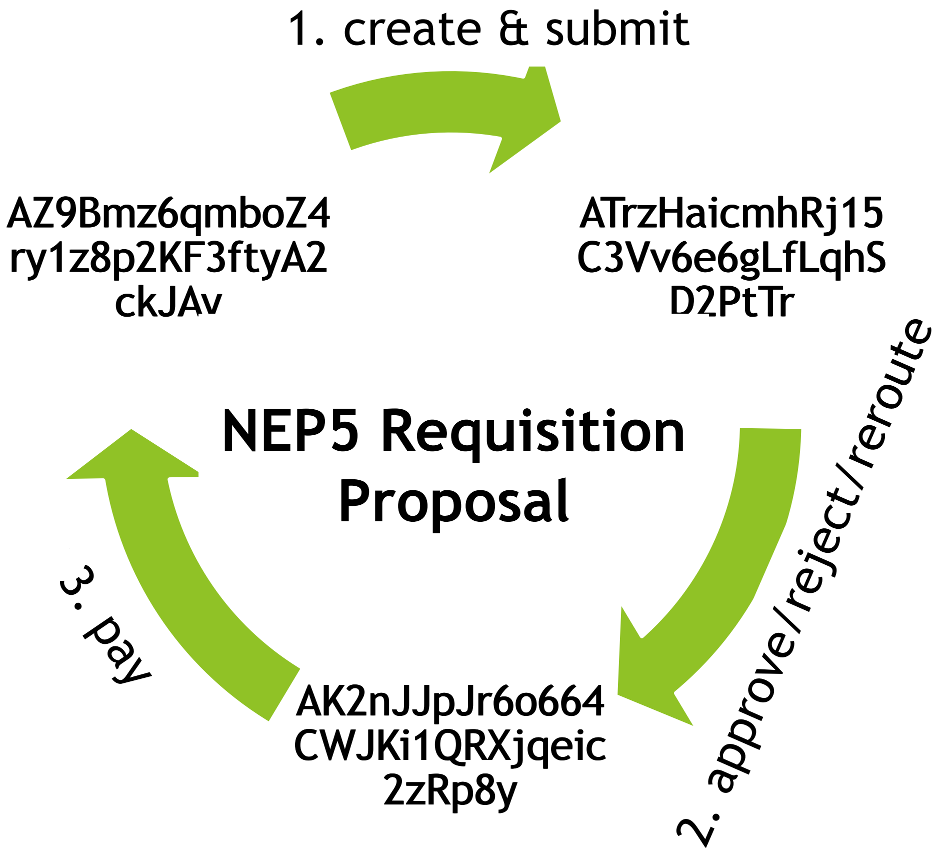


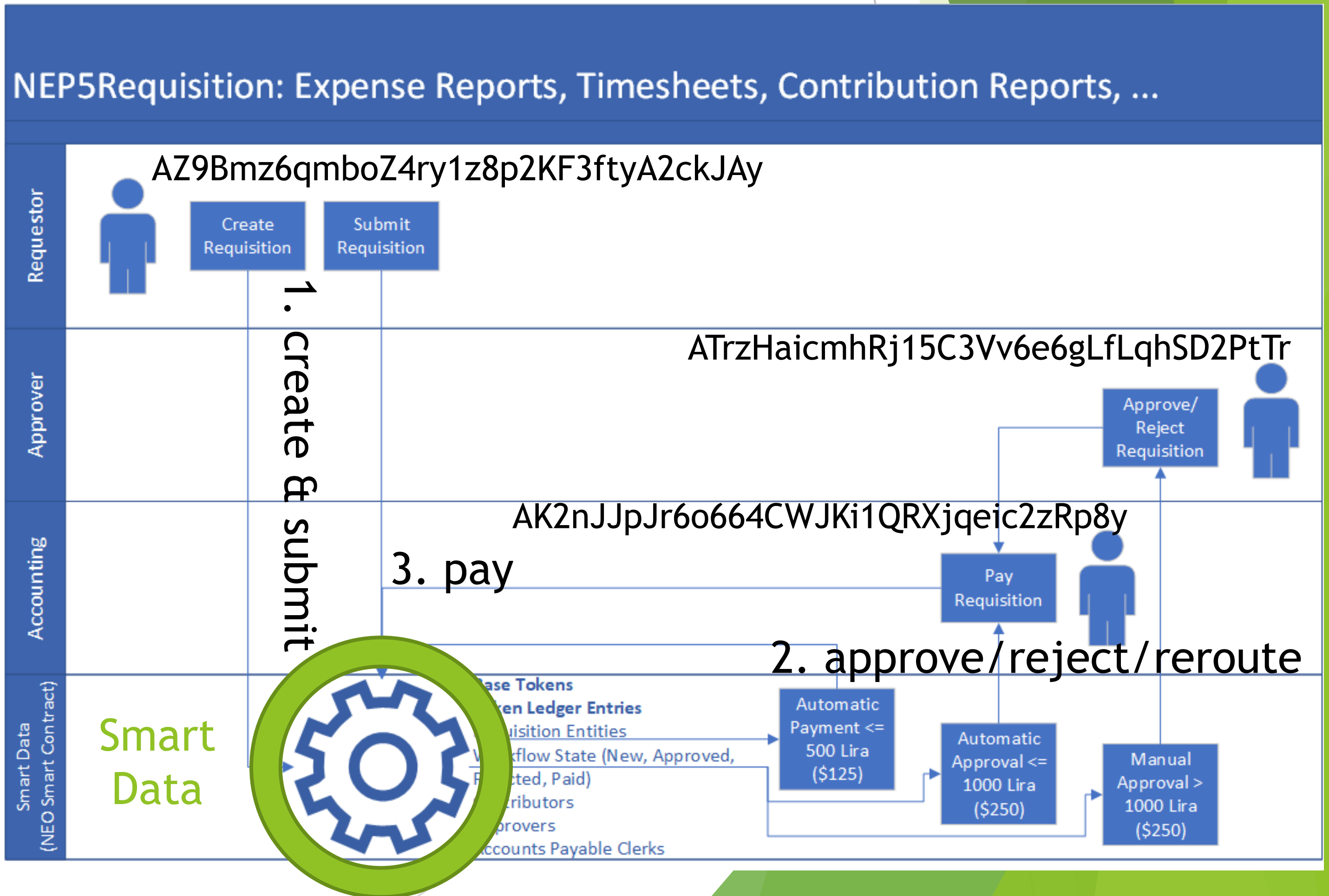
Although some days I feel like this…

NEO dApp Competitions ~USD$500K in prizes
1st NEO Dev Competition
Nov. 20, 2017 - Mar. 31, 2018
NEO
Microsoft
Java
CITY OF ZION IS LAUNCHING ITS SECOND
DAPPS COMPETITION
And we want you to get involved
The City of Zion council is promoting a decentralised apps (dApps) competition to further elevate the NEO developer community.
NEO.NET Developers Center of Excellence
TÜBİTAK
BİLGEM
BLOKZİNCİR
MİCHAEL

Think Deeply about the Smart Economy

"Future world where the boundary between real assets in the physical world and digital assets in the digital world has been removed." [Malcolm Lerider]


1. create & submit
AZ9Bmz6qmboZ4ry1z8p2KF3ftyA2ckJAy
ATrzHaicmhRj15C3Vv6e6gLfLqhSD2PtTr
NEP5 Requisition Proposal
2. approve/reject/reroute
3. pay
AK2nJJpJr6o664CWJKi1QRXjqeic2zRp8y

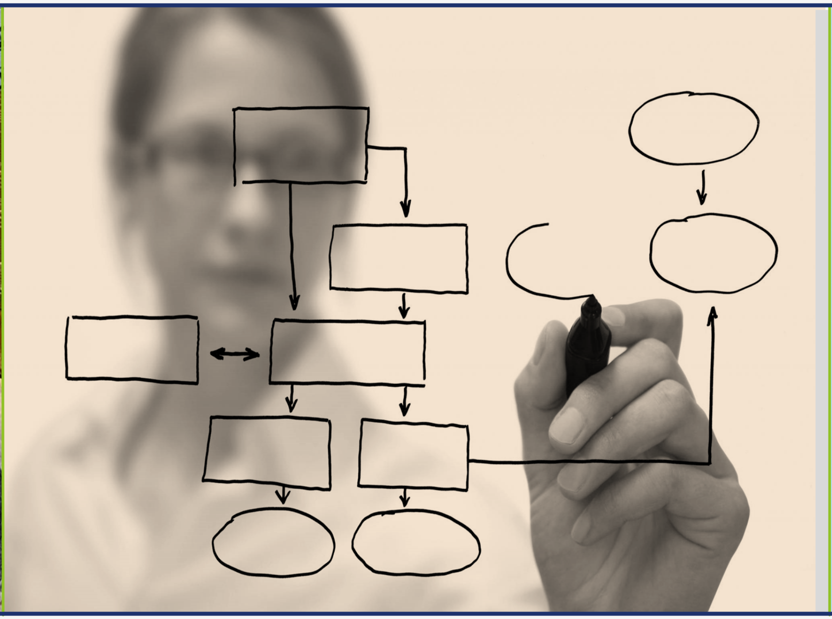

NEP5Requisition: Expense Reports, Timesheets, Contribution Reports, ...
Requestor
AZ9Bmz6qmboZ4ry1z8p2KF3ftyA2ckJAy
Create Requisition
Submit Requisition
1. create & submit
Approver
ATrzHaicmhRj15C3Vv6e6gLfLqhSD2PtTr
Approve/ Reject Requisition
Accounting
AK2nJJpJr6o664CWJKi1QRXjqeic2zRp8y
3. pay
Pay Requisition
2. approve/reject/reroute
Smart Data (NEO Smart Contract)
Smart Data
ase Tokens
ken Ledger Entries
uisition Entities
W kflow State (New, Approved,
R cted, Paid)
ributors
provers
ccounts Payable Clerks
Automatic Payment <= 500 Lira ($125)
Automatic Approval <= 1000 Lira ($250)
Manual Approval > 1000 Lira ($250)

TRUSTED DIGITAL WEB

Trusted
Digital
Web

Fully Integrated Experience

Decentralized Currency
(cryptocurrency)



Decentralized Business
Apps
(business processes)

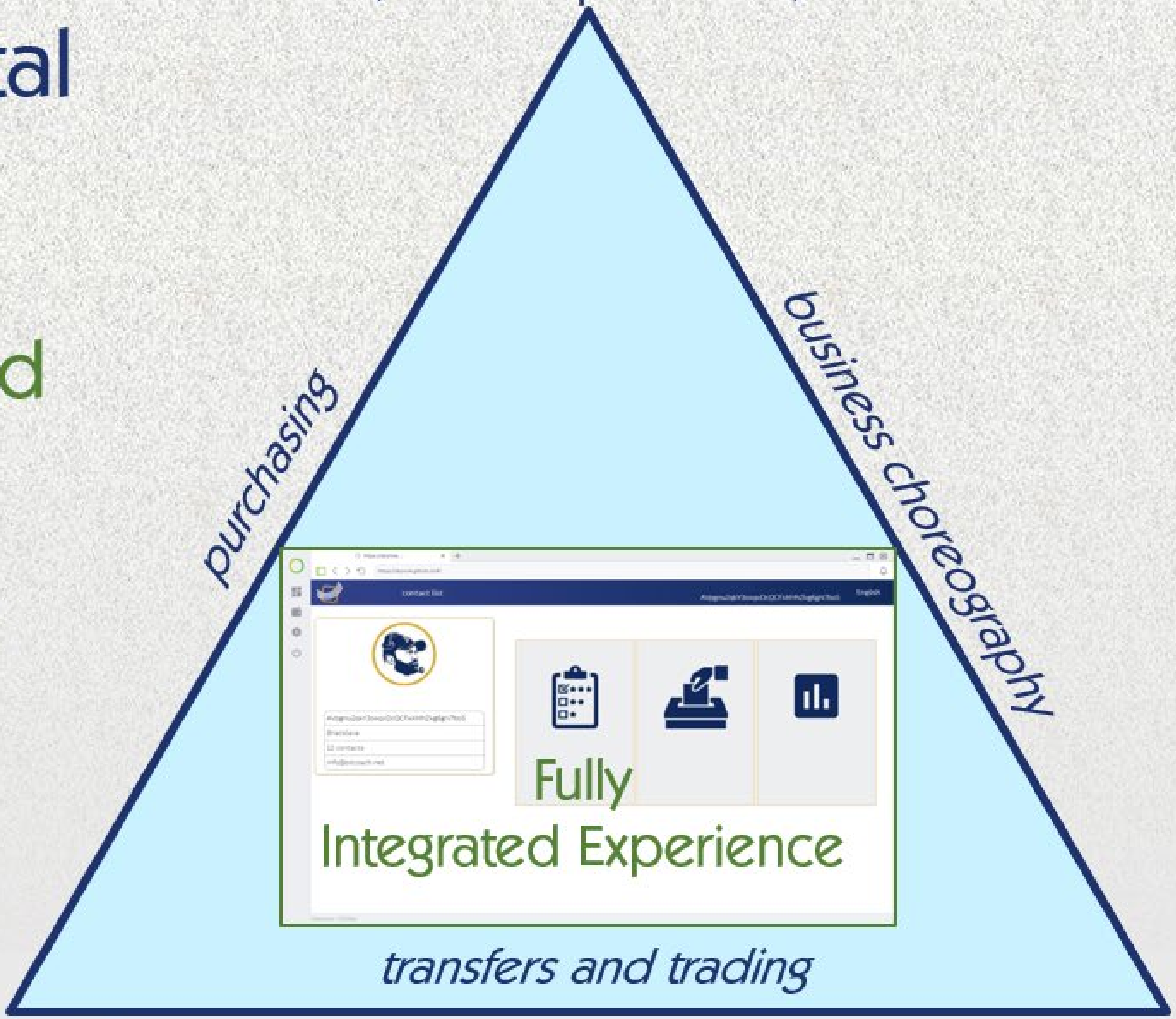
Decentralized Digital
Identity
(people, organizations,
things,
& business documents)

Trusted Digital Web

Fully Integrated Experience

Credit: nOS Project

Business Apps
(business processes)



Fully Integrated Experience

Decentralized Currency
(cryptocurrency)

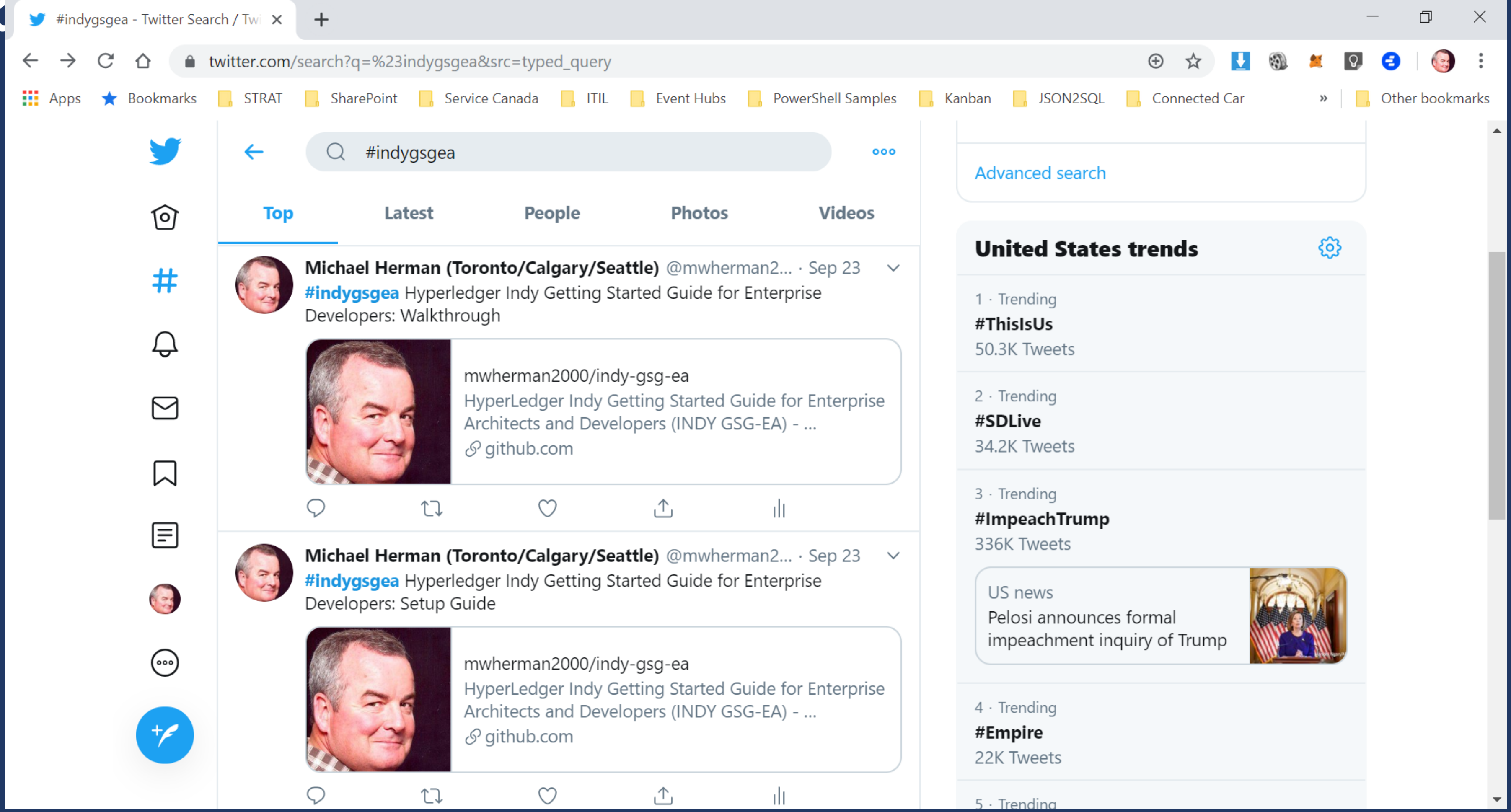
Decentralized Digital Identity
(people, organizations, things,
& business documents)

Training Goals for Today

- Target audience
  - Blockchain Architects and Developers interesting in learning more about Digital Identity
  - “Making software easy to understand – any easy for you to explain to others”
- Level
  - Awareness and Basic Knowledge of Digital Identity concepts
- User Story
  - “Alice buys a car” explained in a way that makes sense for Enterprise Software Developers

Resources

- Can be found by searching Twitter for: #ind



Michael Herman (Toronto/Calgary/Seattle) @mwherman2... · Sep 23
#indygsgea Hyperledger Indy Getting Started Guide for Enterprise Developers: Walkthrough

mwherman2000/indy-gsg-ea
HyperLedger Indy Getting Started Guide for Enterprise Architects and Developers (INDY GSG-EA) - ...
github.com

Michael Herman (Toronto/Calgary/Seattle) @mwherman2... · Sep 23
#indygsgea Hyperledger Indy Getting Started Guide for Enterprise Developers: Setup Guide

mwherman2000/indy-gsg-ea
HyperLedger Indy Getting Started Guide for Enterprise Architects and Developers (INDY GSG-EA) - ...
github.com

United States trends

1 · Trending
#ThisIsUs
50.3K Tweets

2 · Trending
#SDLive
34.2K Tweets

3 · Trending
#ImpeachTrump
336K Tweets

US news
Pelosi announces formal impeachment inquiry of Trump

4 · Trending
#Empire
22K Tweets

5 · Trending

Core Concepts

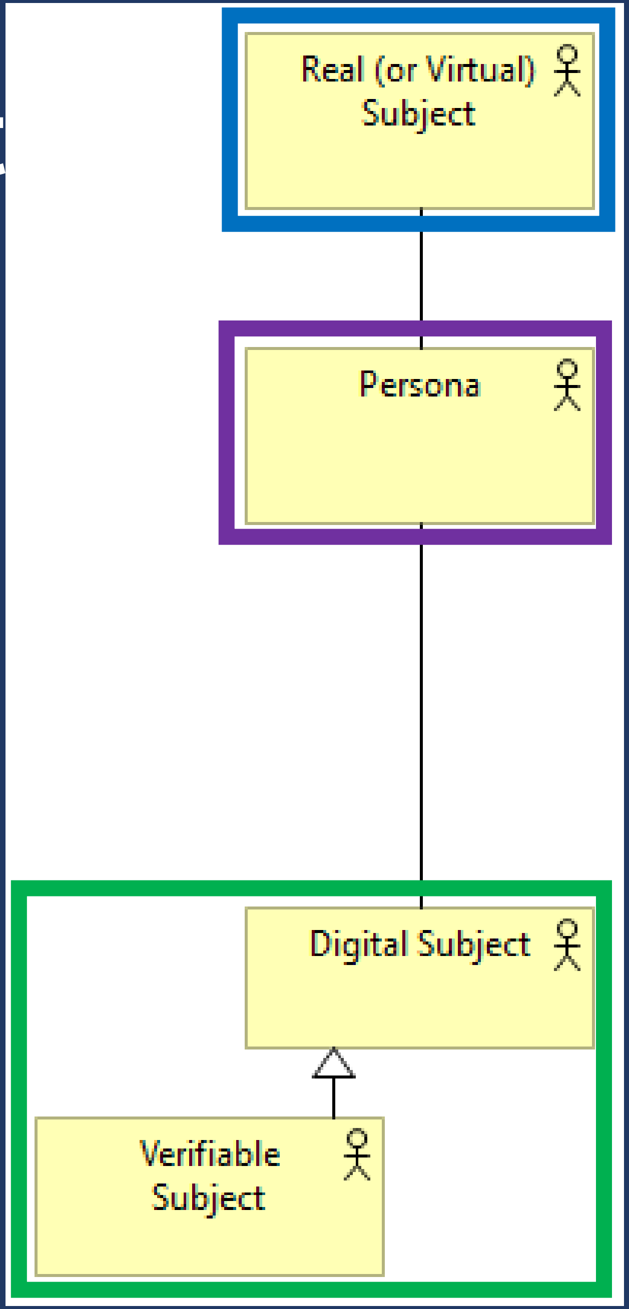
- Subjects
  - Real (or Virtual) Subjects
  - Personas
  - Digital Subject

- Digital Identities
  - Digital Identifiers (DIDs)
  - Digital Identities (Verifiable)
  - Credentials (Verifiable)
  - Claims
  - Profiles
  - Verifiable Subject

Subjects

- Unique Subject
  - A Unique Subject is any unique and specific non-fungible object in the Physical or Digital Universe
    - a person, a place, a thing, an organization, digital visual or audio composition, business document, photo of a piece of toast, etc.
- Digital Subject (Subject)
  - A Digital Subject (aka Subject) is a unique digital representation of a Unique Subject
  - More specifically, a particular Persona of a Unique Subject.

Subject



- Michael Herman
  - The Person

- Self-Sovereign Blockchain Architect
- Self-Sovereign Blockchain Developer
- Alberta Canada Cattle Rancher
- Father

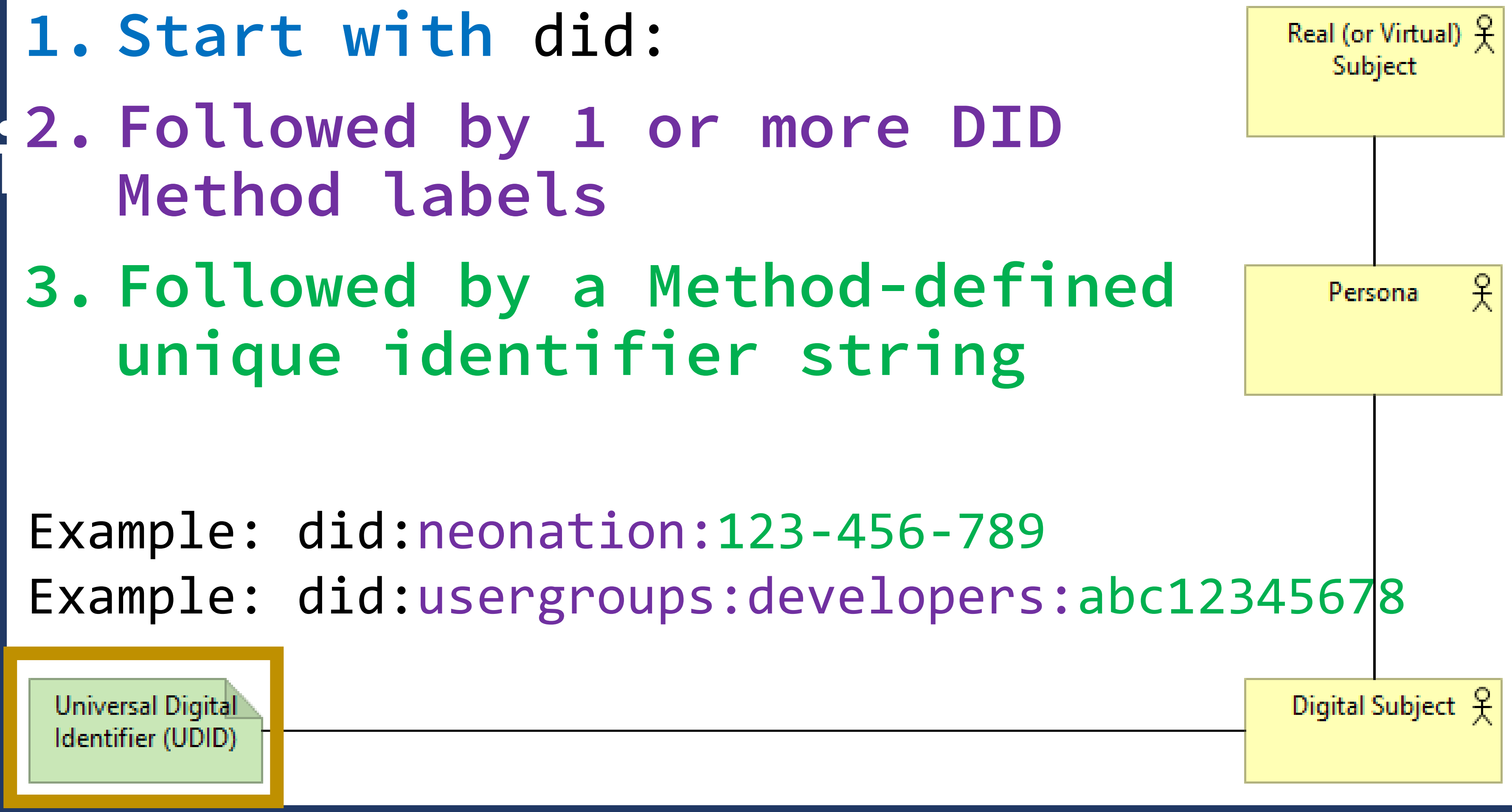
- Farm Producer (Rancher) in Alberta, Canada
- Issued a Alberta Farm Fuel

Digital Identifi
(DIDs)

1. Start with did:
2. Followed by 1 or more DID Method labels
3. Followed by a Method-defined unique identifier string

Example: did:neonation:123-456-789

Example: did:usergroups:developers:abc12345678

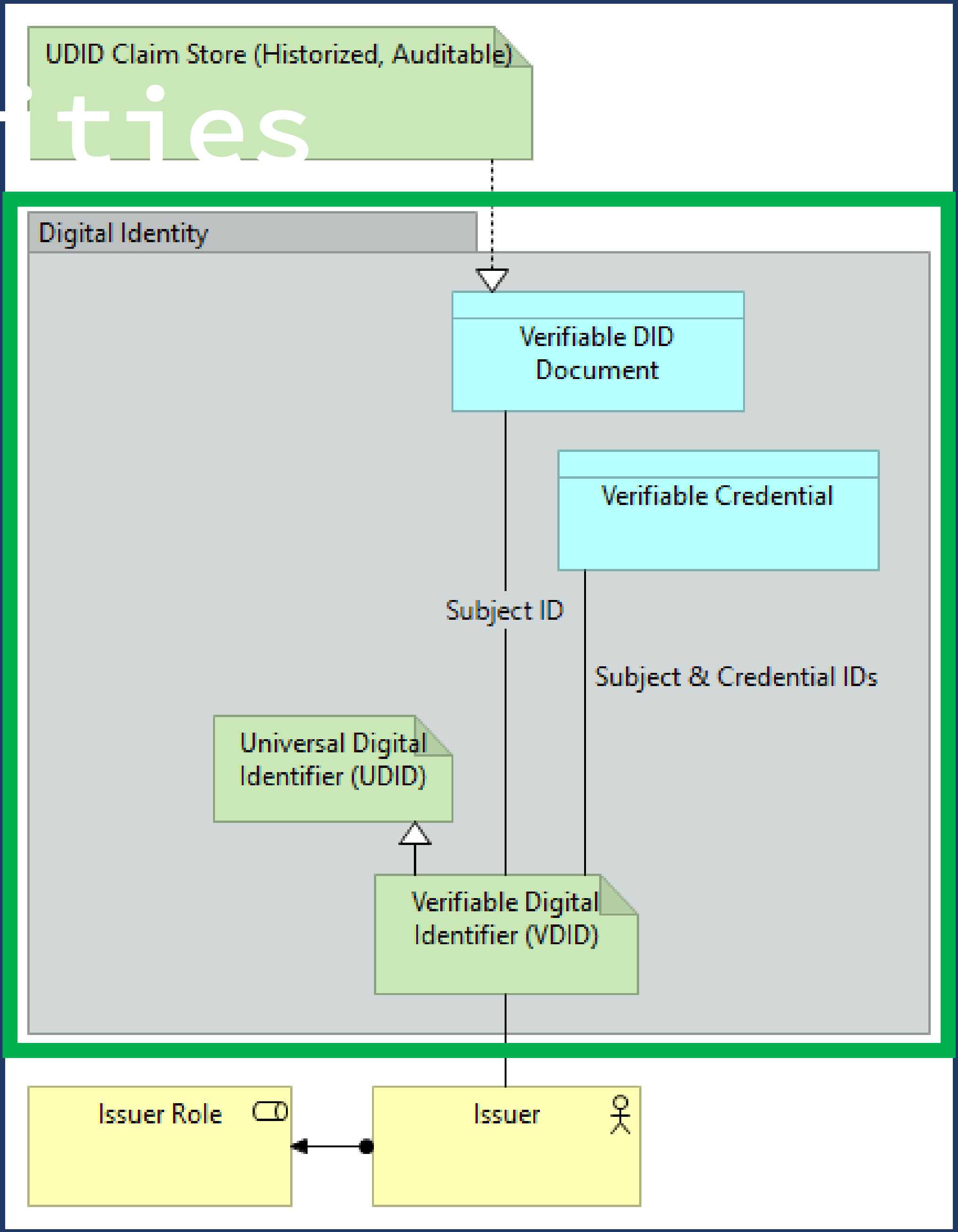

Real (or Virtual) Subject
Persona
Digital Subject
Universal Digital Identifier (UDID)

Digital Identities

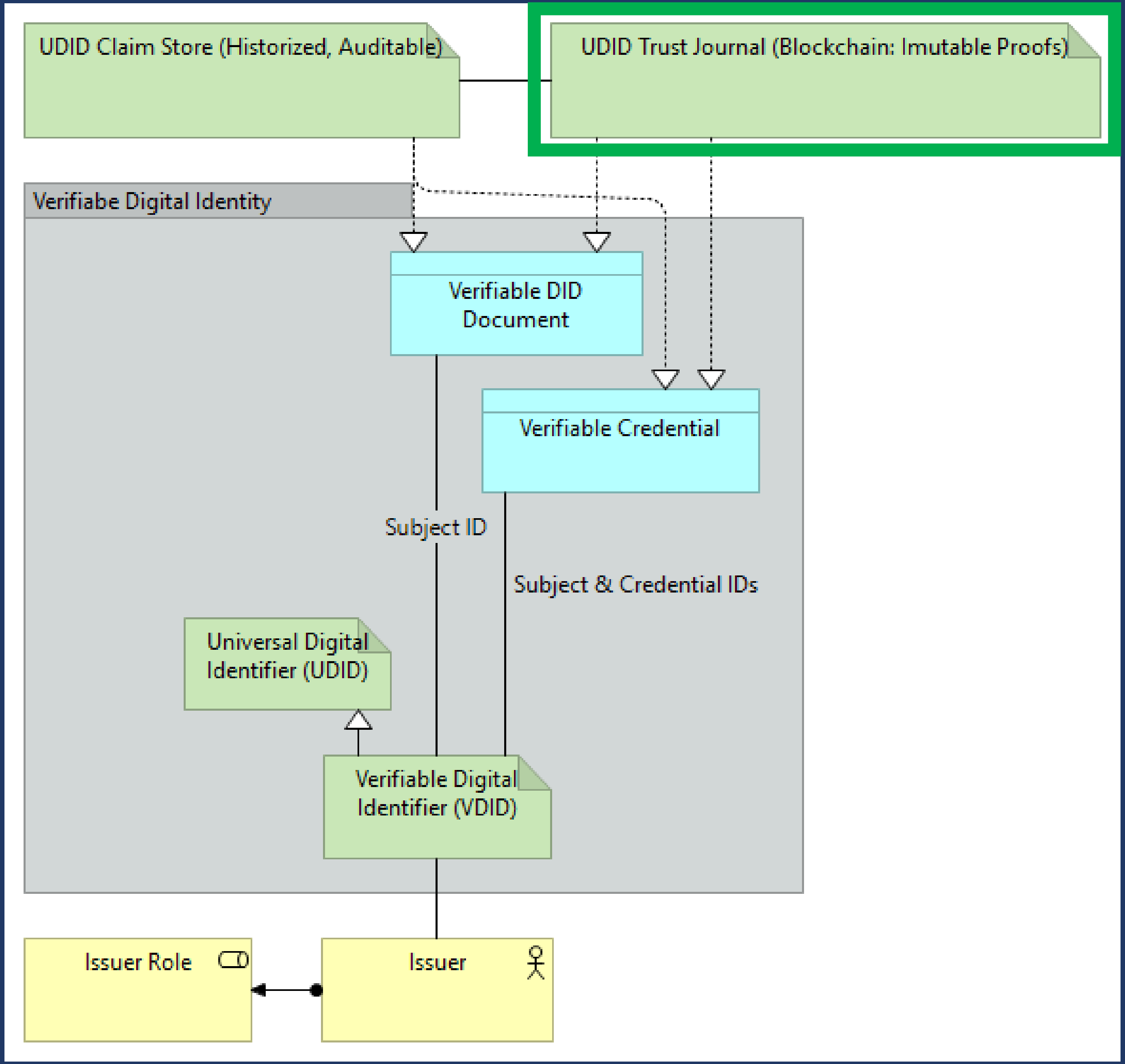
- A Digital Identity is a set of Claims made by one Digital Subject about itself or another Digital Subject [The Laws Of Identity]
  - A Digital Identity is associated with one or more Digital Identifiers (DIDs).

- A Claim is any data associated with a Digital Identity by way of a DID
  - A Claim is a name-value pair representing a datum associated with a DID
  - Preferably, Claim data and the Claims' relationships to a Digital Identity are represented (persisted) in a manner that is:
    - immutable, auditable, verifiable, historized, and permanent

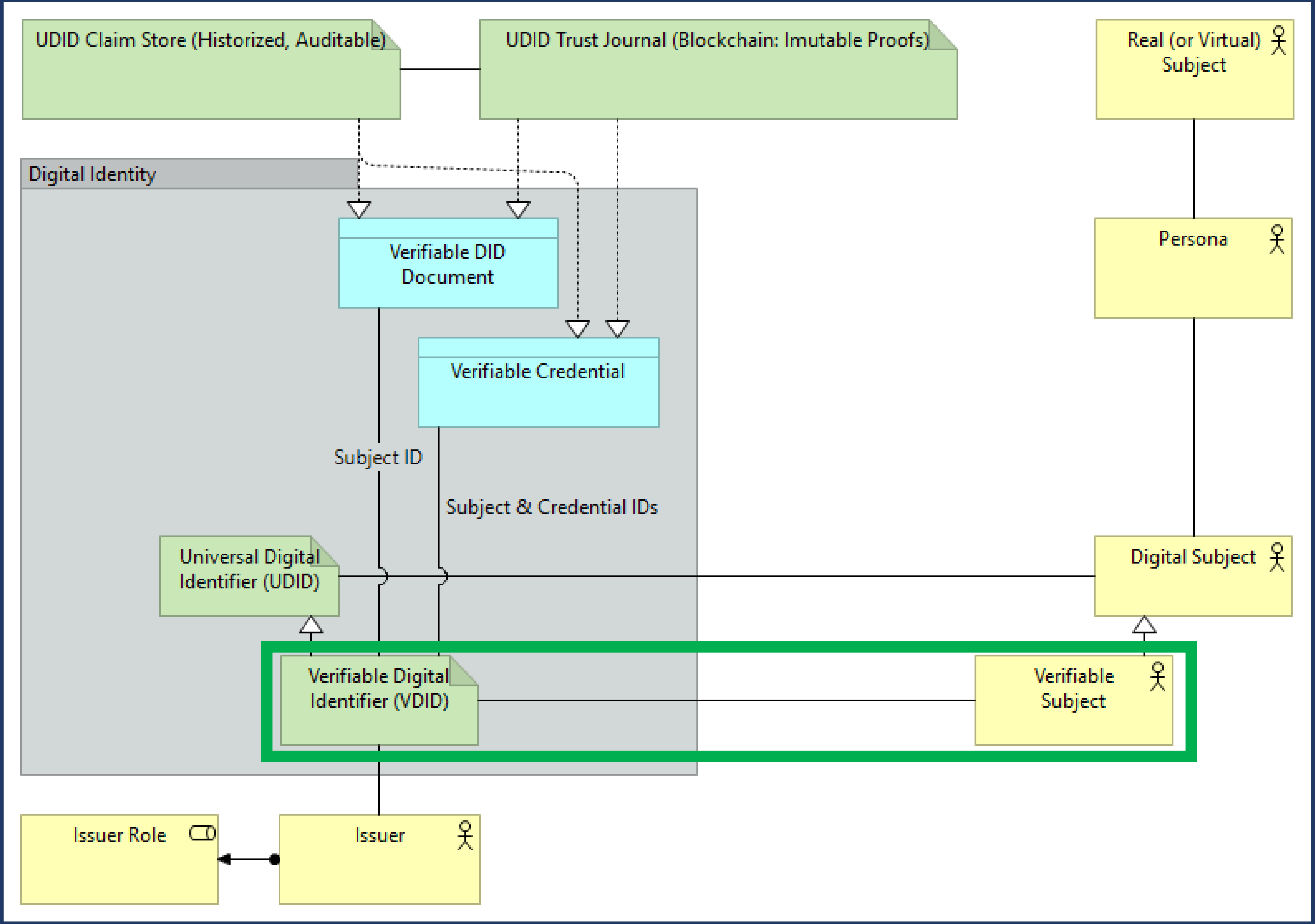
Digital Identities


UDID Claim Store (Historized, Auditable)
Digital Identity
Verifiable DID Document
Verifiable Credential
Subject ID
Subject & Credential IDs
Universal Digital Identifier (UDID)
Verifiable Digital Identifier (VDID)
Issuer Role
Issuer

able Digital Identities


UDID Claim Store (Historized, Auditable)
UDID Trust Journal (Blockchain: Imutable Proofs)
Verifiabe Digital Identity
Verifiable DID Document
Verifiable Credential
Subject ID
Subject & Credential IDs
Universal Digital Identifier (UDID)
Verifiable Digital Identifier (VDID)
Issuer Role
Issuer

Verifia Subject

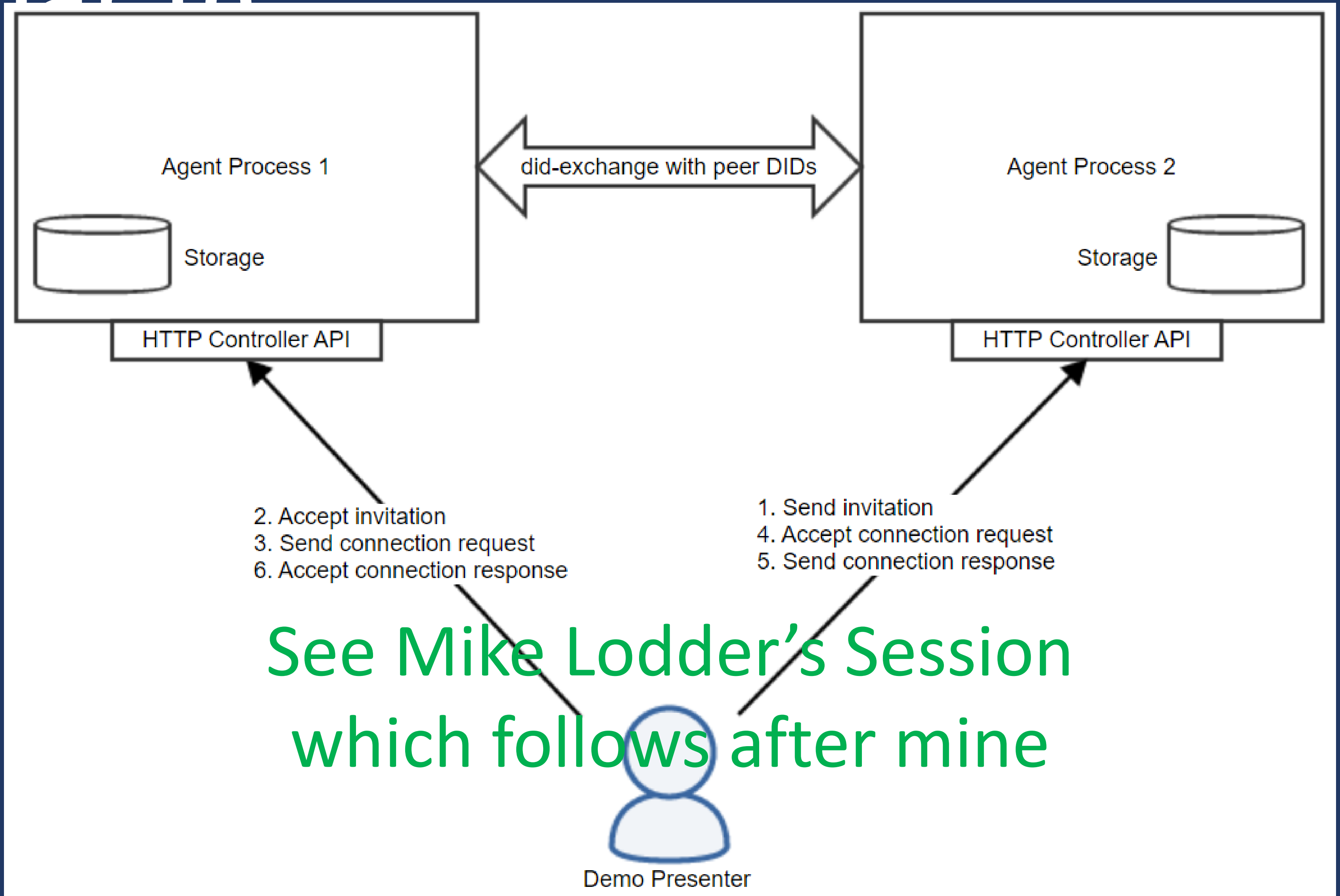


UDID Claim Store (Historized, Auditable)
UDID Trust Journal (Blockchain: Imutable Proofs)
Real (or Virtual) Subject
Digital Identity
Verifiable DID Document
Persona
Verifiable Credential
Subject ID
Subject & Credential IDs
Universal Digital Identifier (UDID)
Digital Subject
Verifiable Digital Identifier (VDID)
Verifiable Subject
Issuer Role
Issuer

Credit: Lunch with Marc Aniballi / Toronto

Hyperledger Indy Getting Started Guide for Enterprise Architects and Developers

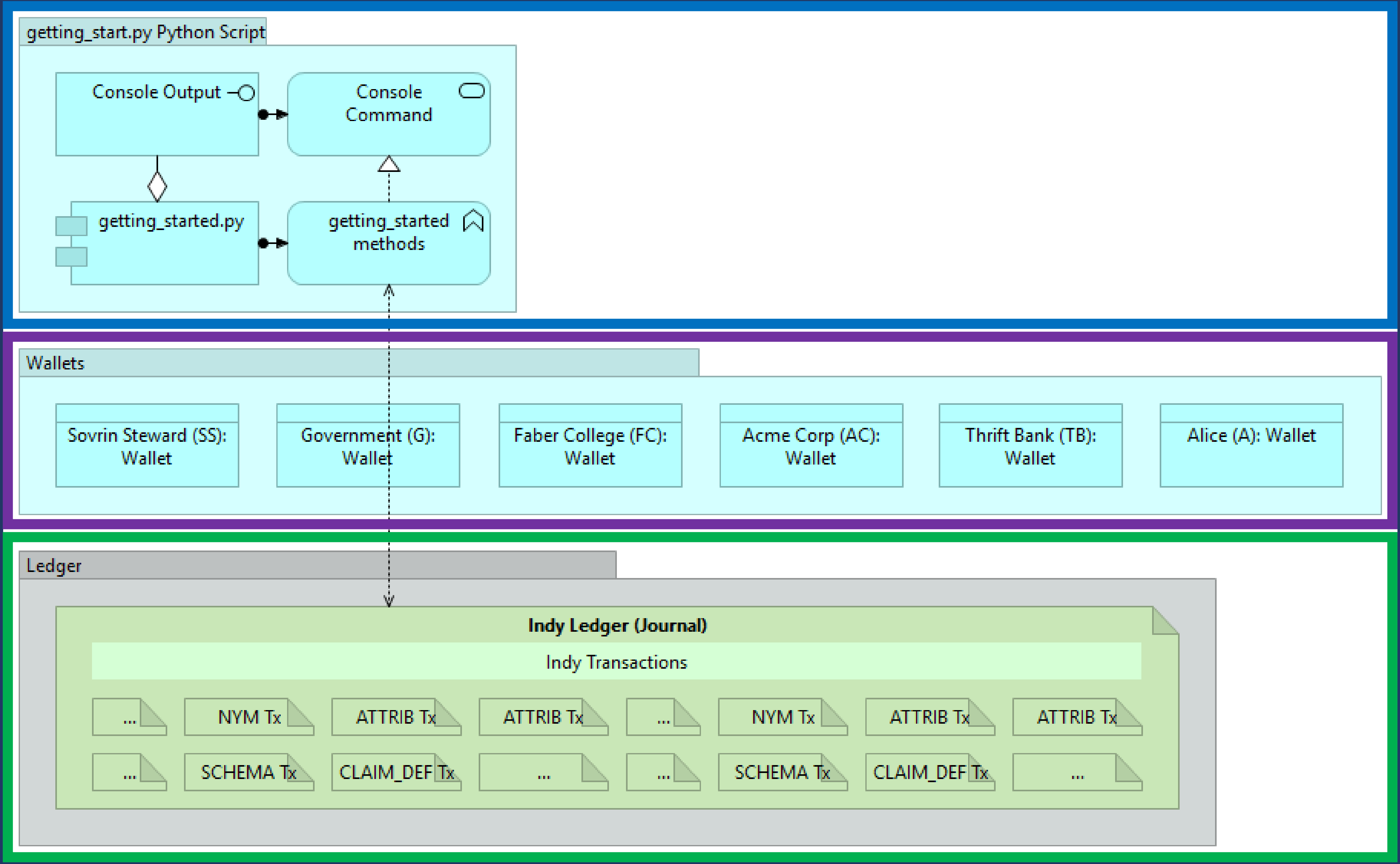
Normal Indy-Aries Demo/Training Scenario



2. Accept invitation
3. Send connection request
6. Accept connection response

1. Send invitation
4. Accept connection request
5. Send connection response

See Mike Lodder’s Session which follows after mine



getting_start.py Python Script

Console Output

Console Command

getting_started.py

getting_started methods

Wallets

Sovrin Steward (SS): Wallet

Government (G): Wallet

Faber College (FC): Wallet

Acme Corp (AC): Wallet

Thrift Bank (TB): Wallet

Alice (A): Wallet

Ledger

Indy Ledger (Journal)

Indy Transactions

...

NYM Tx

ATTRIB Tx

ATTRIB Tx

...

NYM Tx

ATTRIB Tx

ATTRIB Tx

...

SCHEMA Tx

CLAIM_DEF Tx

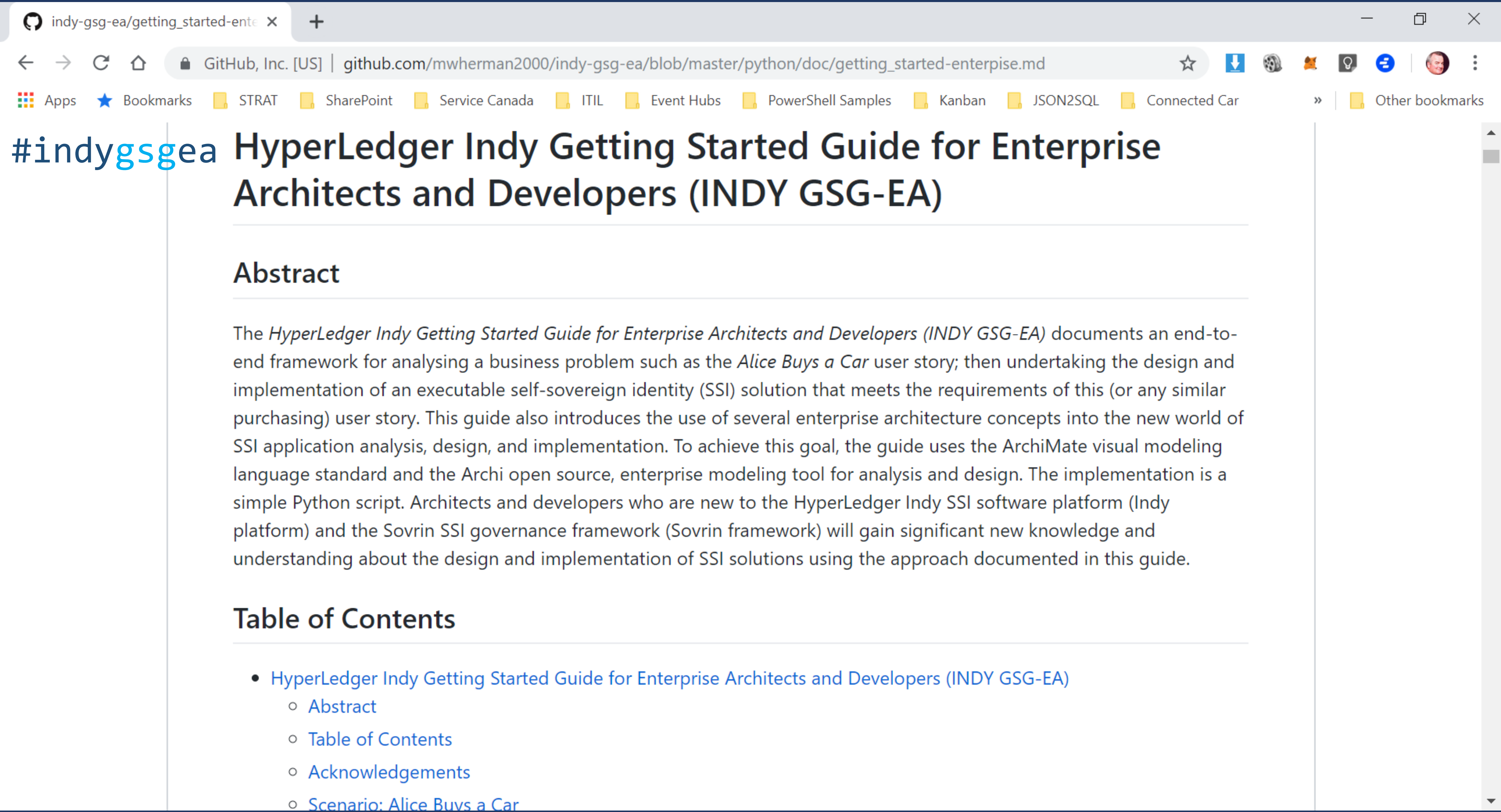
...

...

SCHEMA Tx

CLAIM_DEF Tx

...


indy-gsg-ea/getting_started-ente
GitHub, Inc. [US] | github.com/mwherman2000/indy-gsg-ea/blob/master/python/doc/getting_started-enterpise.md
Apps
Bookmarks
STRAT
SharePoint
Service Canada
ITIL
Event Hubs
PowerShell Samples
Kanban
JSON2SQL
Connected Car
Other bookmarks

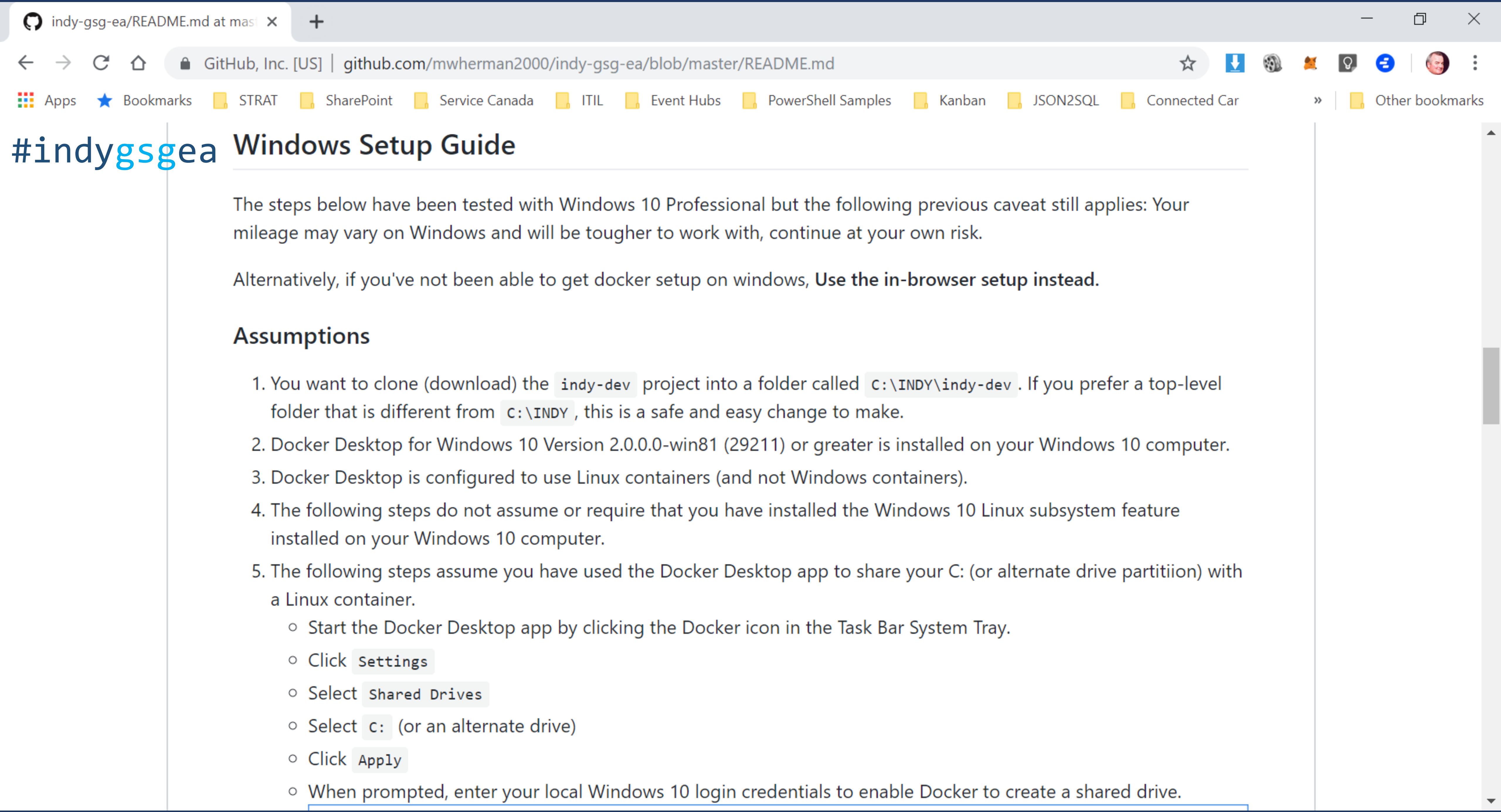
#indygsgea

HyperLedger Indy Getting Started Guide for Enterprise Architects and Developers (INDY GSG-EA)

Abstract

The *HyperLedger Indy Getting Started Guide for Enterprise Architects and Developers (INDY GSG-EA)* documents an end-to-end framework for analysing a business problem such as the *Alice Buys a Car* user story; then undertaking the design and implementation of an executable self-sovereign identity (SSI) solution that meets the requirements of this (or any similar purchasing) user story. This guide also introduces the use of several enterprise architecture concepts into the new world of SSI application analysis, design, and implementation. To achieve this goal, the guide uses the ArchiMate visual modeling language standard and the Archi open source, enterprise modeling tool for analysis and design. The implementation is a simple Python script. Architects and developers who are new to the HyperLedger Indy SSI software platform (Indy platform) and the Sovrin SSI governance framework (Sovrin framework) will gain significant new knowledge and understanding about the design and implementation of SSI solutions using the approach documented in this guide.

Table of Contents



#indygsgea

Windows Setup Guide

The steps below have been tested with Windows 10 Professional but the following previous caveat still applies: Your mileage may vary on Windows and will be tougher to work with, continue at your own risk.

Alternatively, if you've not been able to get docker setup on windows, **Use the in-browser setup instead.**

Assumptions

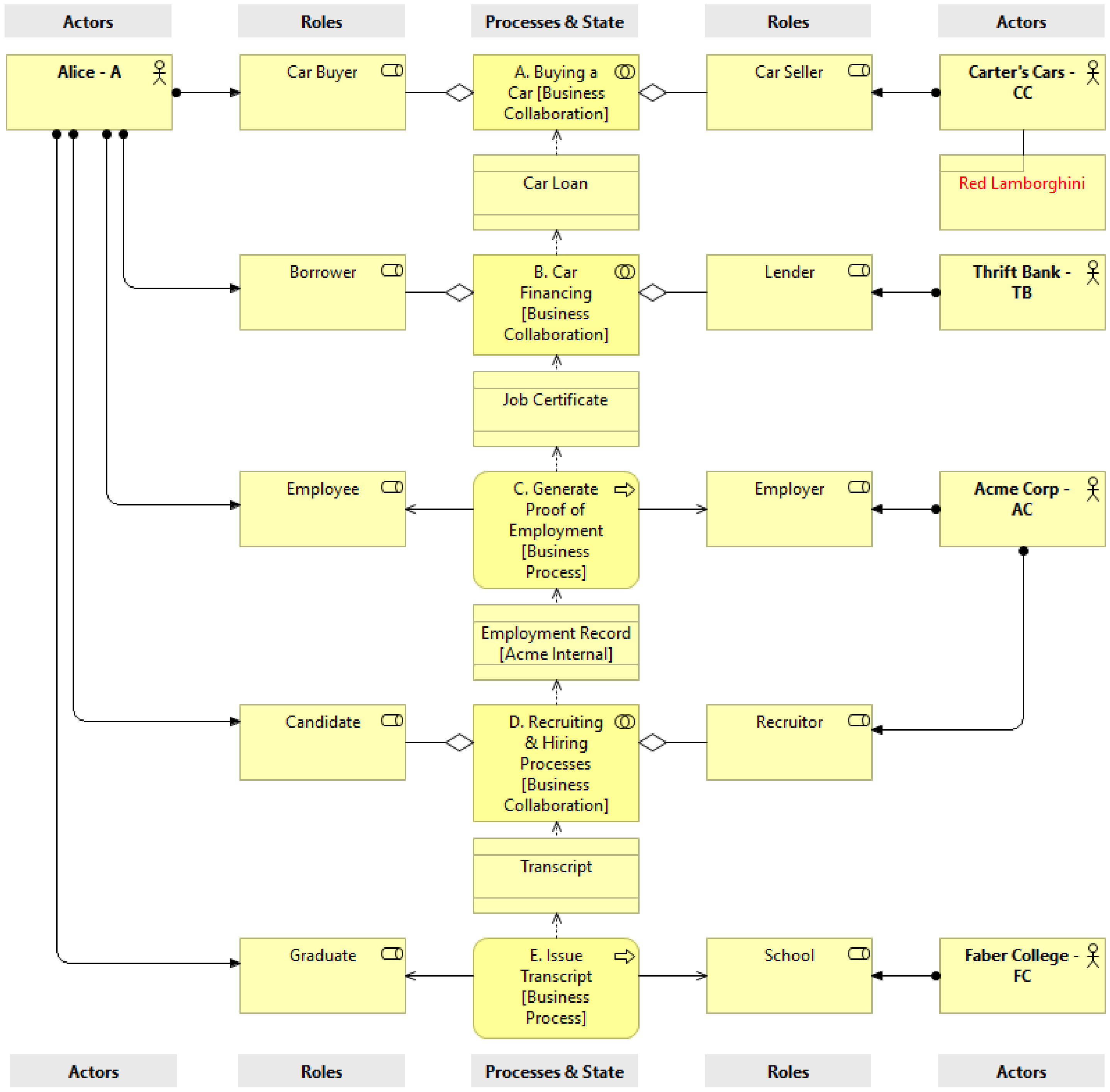
1. You want to clone (download) the `indy-dev` project into a folder called `C:\INDY\indy-dev`. If you prefer a top-level folder that is different from `C:\INDY`, this is a safe and easy change to make.
2. Docker Desktop for Windows 10 Version 2.0.0.0-win81 (29211) or greater is installed on your Windows 10 computer.
3. Docker Desktop is configured to use Linux containers (and not Windows containers).
4. The following steps do not assume or require that you have installed the Windows 10 Linux subsystem feature installed on your Windows 10 computer.
5. The following steps assume you have used the Docker Desktop app to share your C: (or alternate drive partitiion) with a Linux container.
   - Start the Docker Desktop app by clicking the Docker icon in the Task Bar System Tray.
   - Click `Settings`
   - Select `Shared Drives`
   - Select `C:` (or an alternate drive)
   - Click `Apply`
   - When prompted, enter your local Windows 10 login credentials to enable Docker to create a shared drive.

INDY-GSG-EA User Story: Four Use Cases

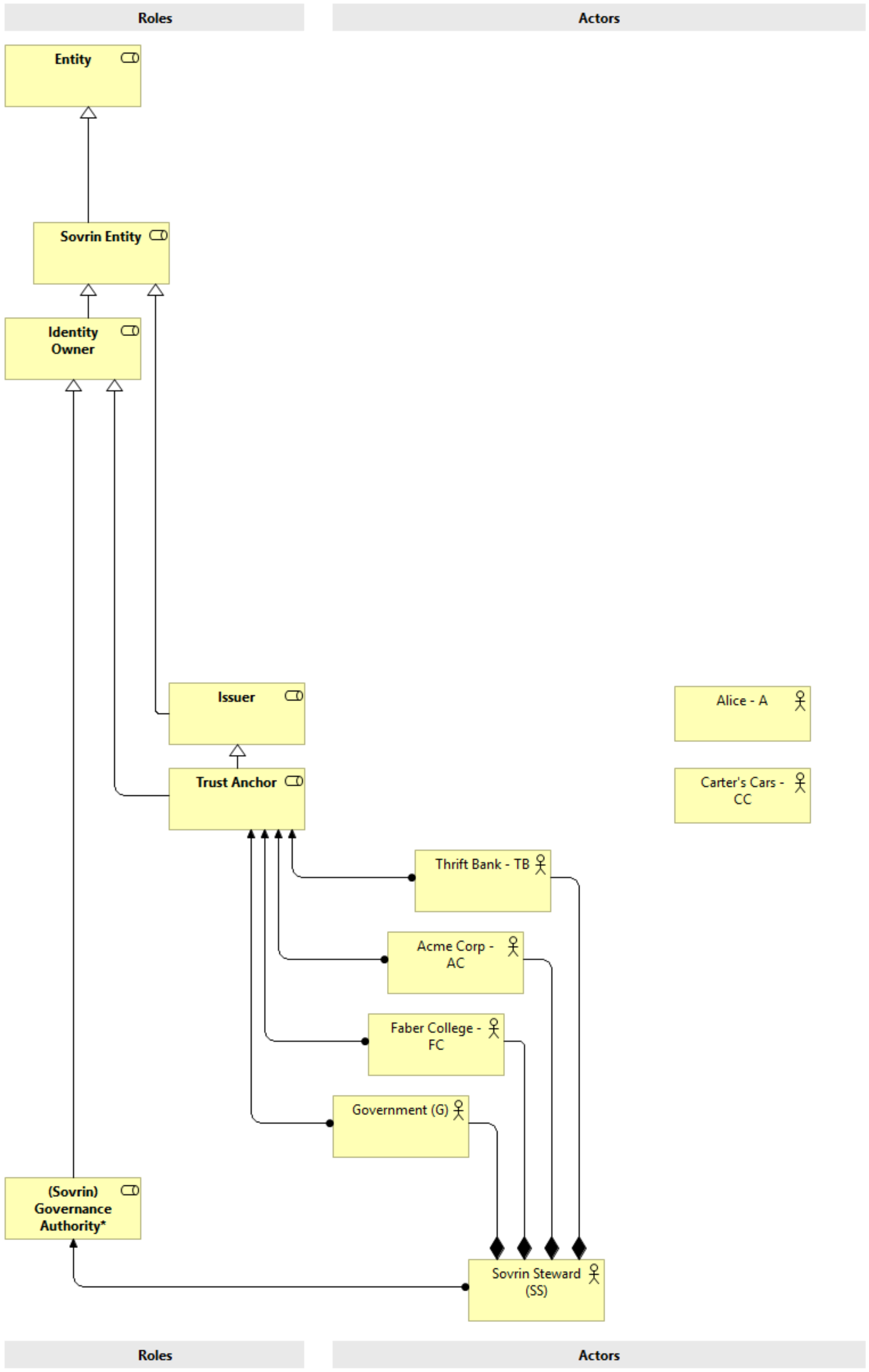
1. Alice wants to buy a car - a bright red Lamborghini® - from Carter's Cars.
2. To purchase the car from Carter's Cars, Alice needs to take out a loan from Thrift Bank for the entire purchase price.
3. To get the loan from Thrift Bank, Alice needs to find a job (Alice is currently unemployed) and present a job certificate from her employer to Thrift Bank verifying her employment, salary level, length of service, etc.
4. To find a job, Alice applies as a candidate for a position at Acme Corp. To apply for a job at Acme Corp, Alice needs to include a verifiable copy of her transcript from Faber

Workflow

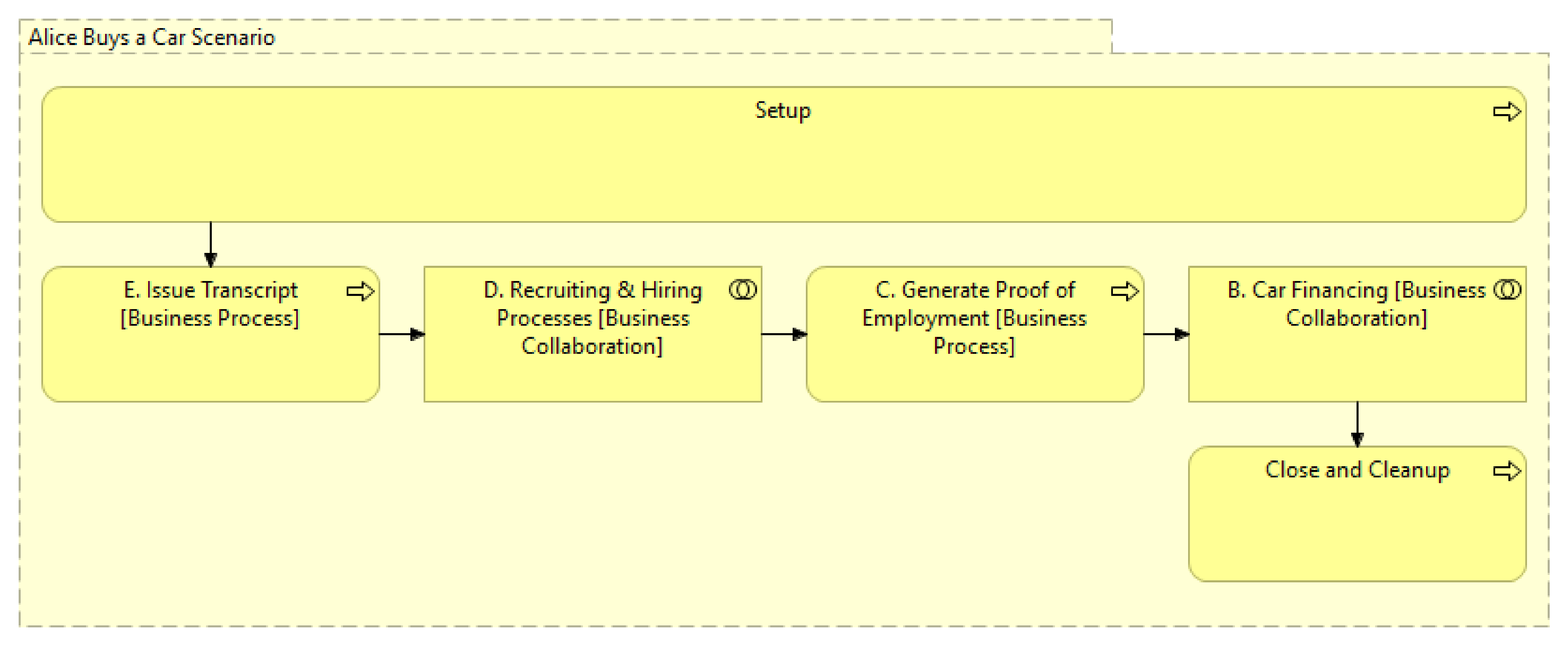
1. Actors
2. Business Roles
3. Processes and Subprocesses
4. State (Artifacts)
   - Business documents represents a “credentials”

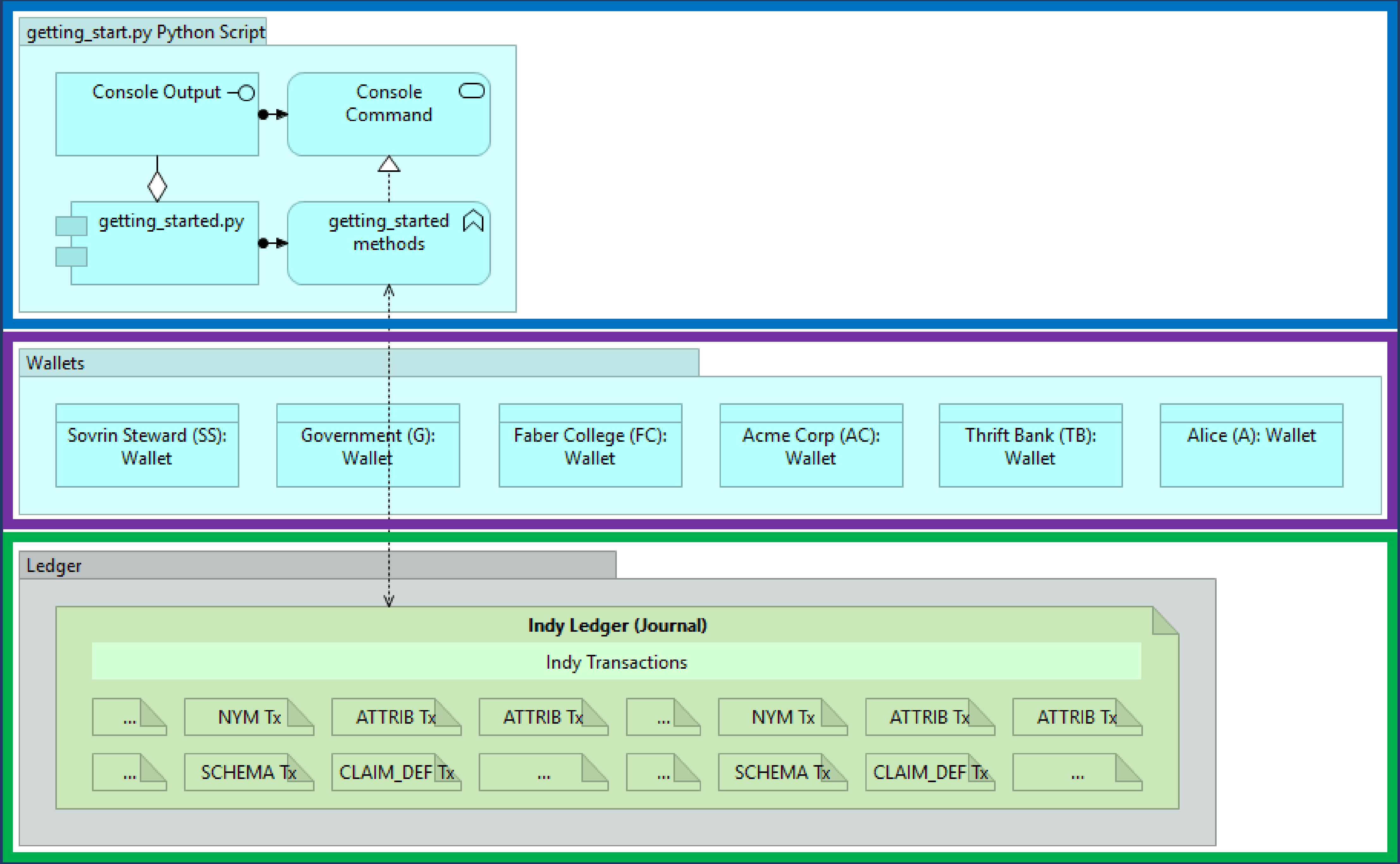


Actors


Roles
Actors
Entity
Sovrin Entity
Identity Owner
Issuer
Trust Anchor
(Sovrin) Governance Authority*
Alice - A
Carter's Cars - CC
Thrift Bank - TB
Acme Corp - AC
Faber College - FC
Government (G)
Sovrin Steward (SS)
Roles
Actors

Business Process Analysis


Alice Buys a Car Scenario
Setup
E. Issue Transcript [Business Process]
D. Recruiting & Hiring Processes [Business Collaboration]
C. Generate Proof of Employment [Business Process]
B. Car Financing [Business Collaboration]
Close and Cleanup



getting_start.py Python Script

Console Output

Console Command

getting_started.py

getting_started methods

Wallets

Sovrin Steward (SS): Wallet

Government (G): Wallet

Faber College (FC): Wallet

Acme Corp (AC): Wallet

Thrift Bank (TB): Wallet

Alice (A): Wallet

Ledger

Indy Ledger (Journal)

Indy Transactions

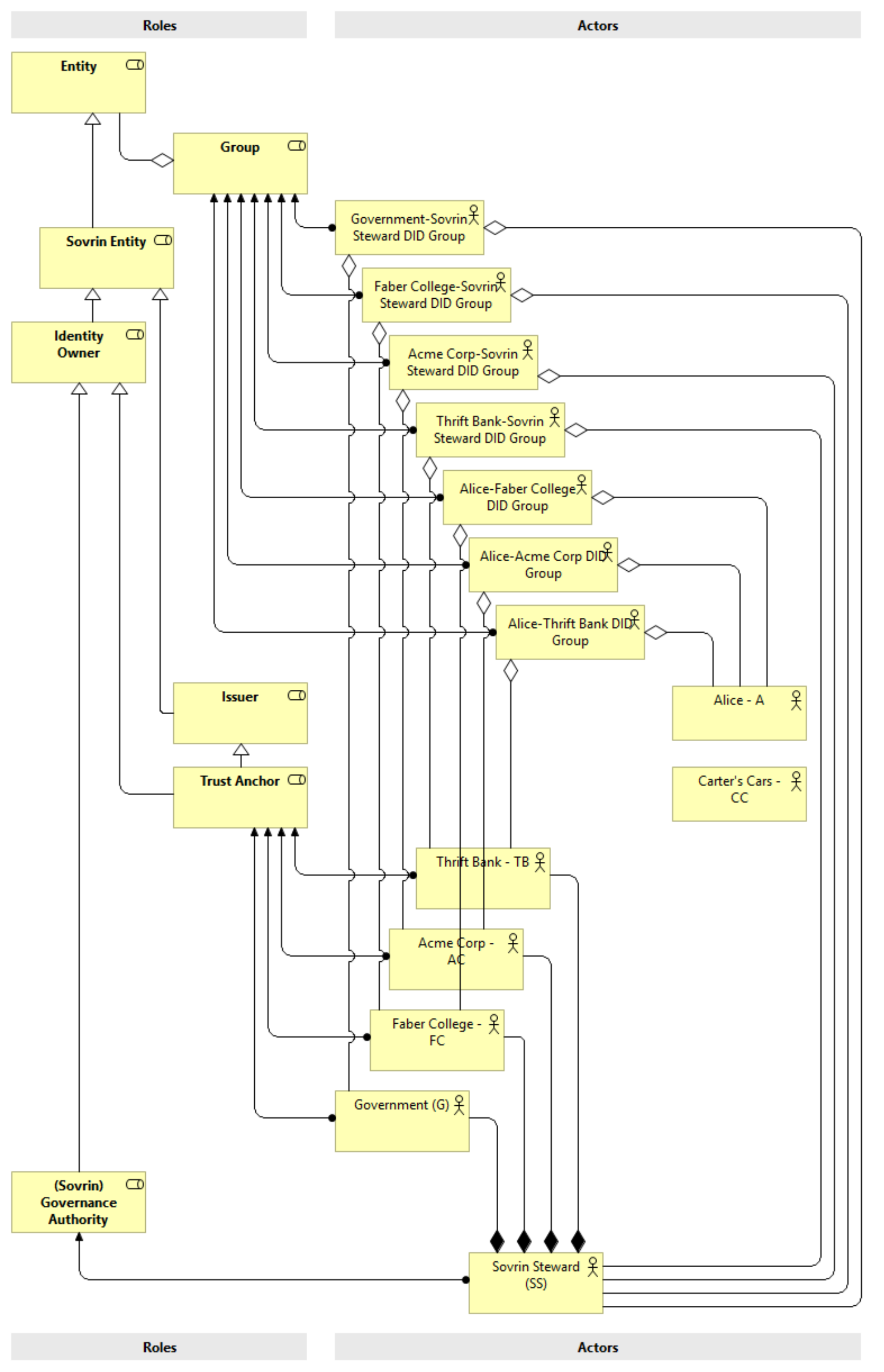
... NYM Tx ATTRIB Tx ATTRIB Tx ... NYM Tx ATTRIB Tx ATTRIB Tx

... SCHEMA Tx CLAIM_DEF Tx ... ... SCHEMA Tx CLAIM_DEF Tx ...

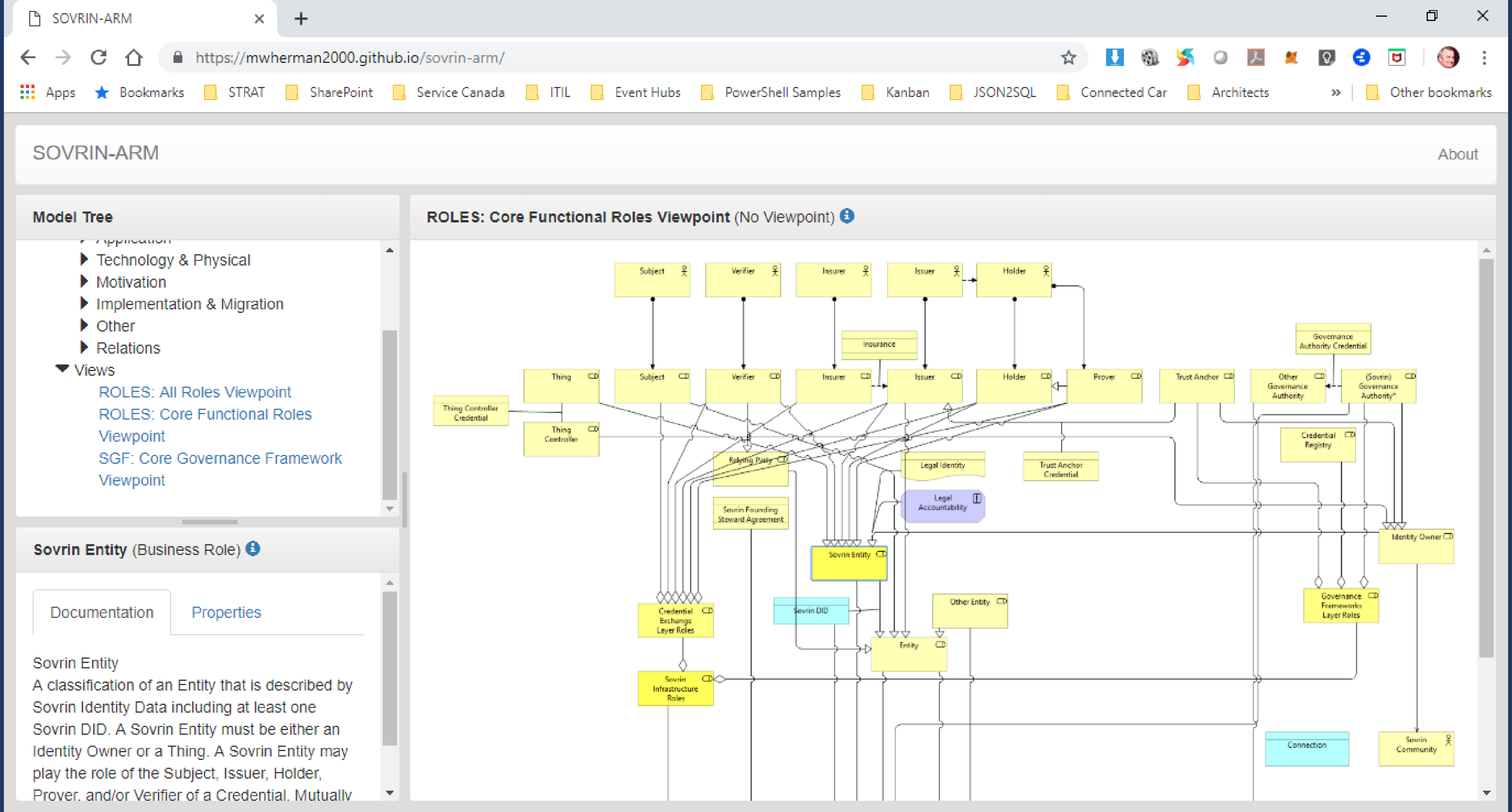
Data Model

- Roles
- Groups
- Actors

Based on Sovrin Gloss


Roles
Actors
Entity
Group
Sovrin Entity
Identity Owner
Government-Sovrin Steward DID Group
Faber College-Sovrin Steward DID Group
Acme Corp-Sovrin Steward DID Group
Thrift Bank-Sovrin Steward DID Group
Alice-Faber College DID Group
Alice-Acme Corp DID Group
Alice-Thrift Bank DID Group
Alice - A
Carter's Cars - CC
Issuer
Trust Anchor
Thrift Bank - TB
Acme Corp - AC
Faber College - FC
Government (G)
(Sovrin) Governance Authority
Sovrin Steward (SS)
Roles
Actors

Sovrin Glossary and Sovrin ARM (#indygsgea)



SOVRIN-ARM

About

Model Tree

- Application
- Technology & Physical
- Motivation
- Implementation & Migration
- Other
- Relations
- Views
  - ROLES: All Roles Viewpoint
  - ROLES: Core Functional Roles Viewpoint
  - SGF: Core Governance Framework Viewpoint

ROLES: Core Functional Roles Viewpoint (No Viewpoint)

Sovrin Entity (Business Role)

Documentation | Properties

Sovrin Entity

A classification of an Entity that is described by Sovrin Identity Data including at least one Sovrin DID. A Sovrin Entity must be either an Identity Owner or a Thing. A Sovrin Entity may play the role of the Subject, Issuer, Holder, Prover, and/or Verifier of a Credential. Mutually

Walkthrough

https://github.com/mwherman2000/indy-gsg-ea/blob/master/python/doc/getting_started-enterpise.md

Indy Getting Started Guide for Enterprise Architects and Developers (INDY GSG-EA) Value Add

- Tested setup and execution process
- Four highly-correlated artifacts
  1. getting_started-numbered.py Script
  2. Numbered Trace File
  3. Communication Diagrams
  4. Getting Started Guide (INDY GSG-EA)

Original getting_started.py

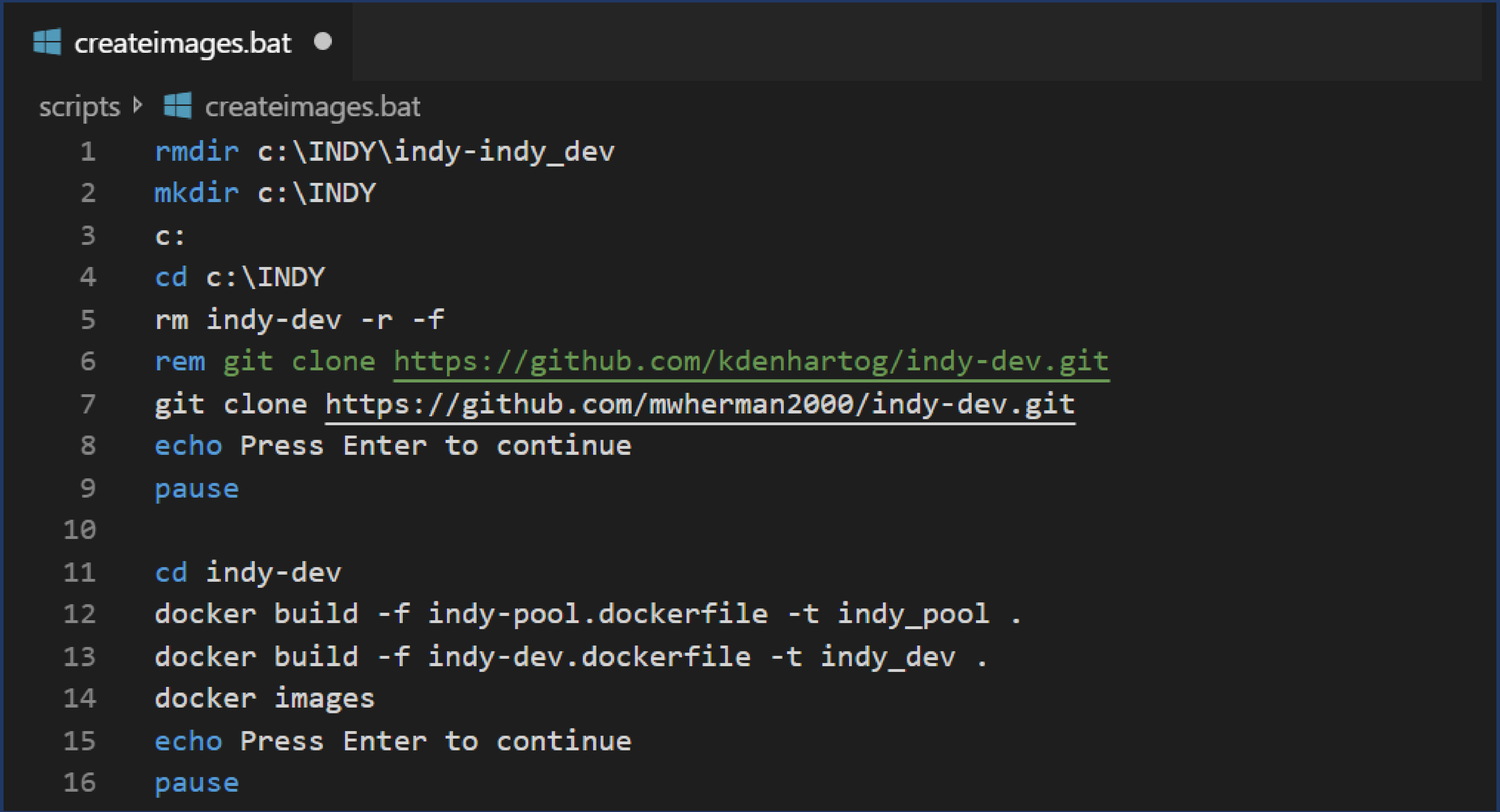
getting_started-numbered.py Script

Numbered Trace File

Communic-ation Diagrams

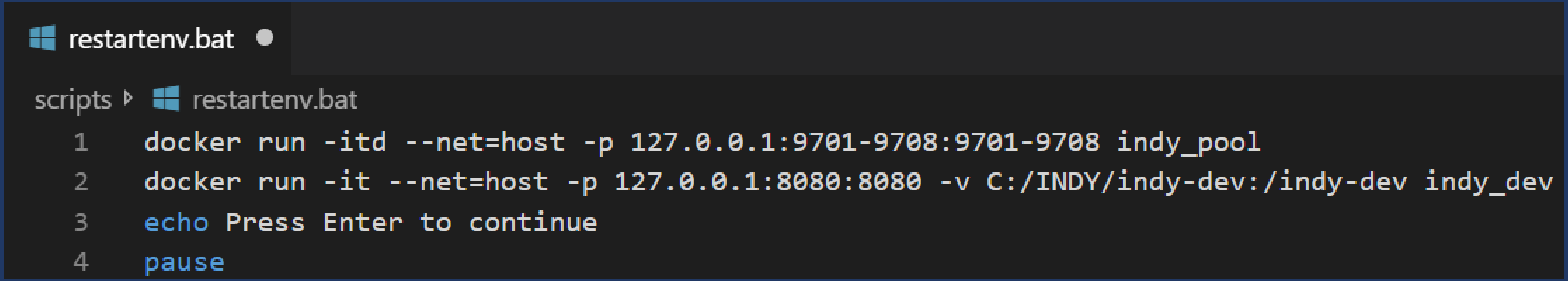
Getting Started Guide (INDY GSG-EA)

Setup 1/2



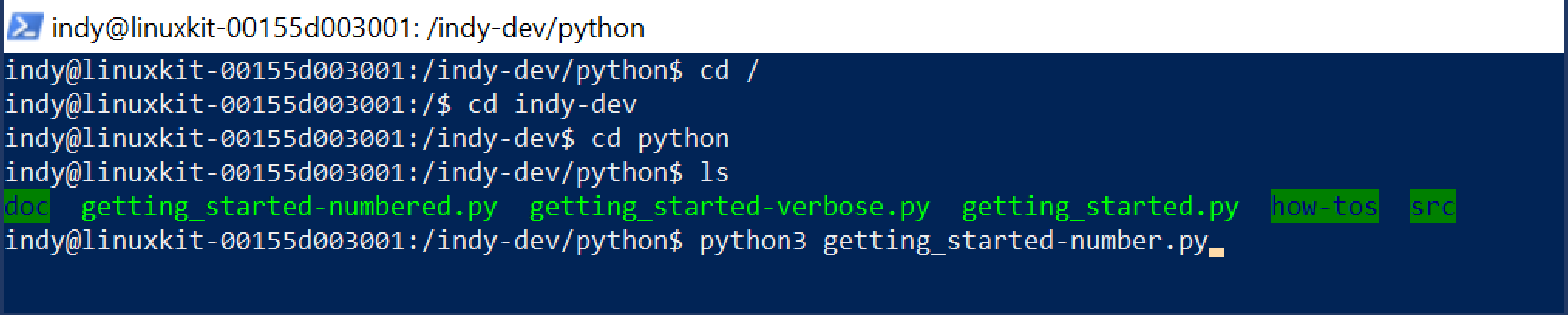
```
rmdir c:\INDY\indy-indy_dev
mkdir c:\INDY
c:
cd c:\INDY
rm indy-dev -r -f
rem git clone https://github.com/kdenhartog/indy-dev.git
git clone https://github.com/mwherman2000/indy-dev.git
echo Press Enter to continue
pause

cd indy-dev
docker build -f indy-pool.dockerfile -t indy_pool .
docker build -f indy-dev.dockerfile -t indy_dev .
docker images
echo Press Enter to continue
pause
```

Setup 2/2



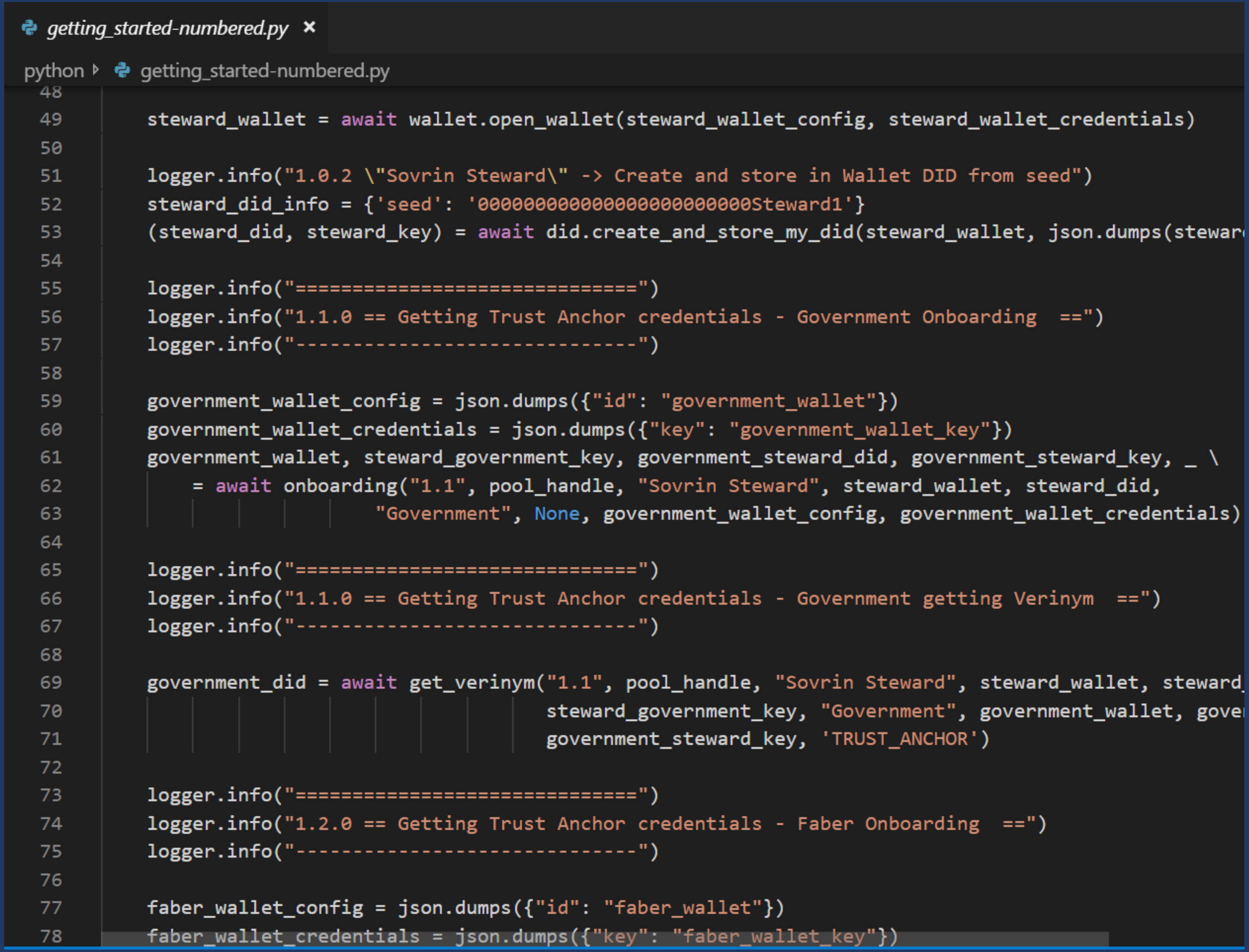
```
docker run -itd --net=host -p 127.0.0.1:9701-9708:9701-9708 indy_pool
docker run -it --net=host -p 127.0.0.1:8080:8080 -v C:/INDY/indy-dev:/indy-dev indy_dev
echo Press Enter to continue
pause
```

Running the script



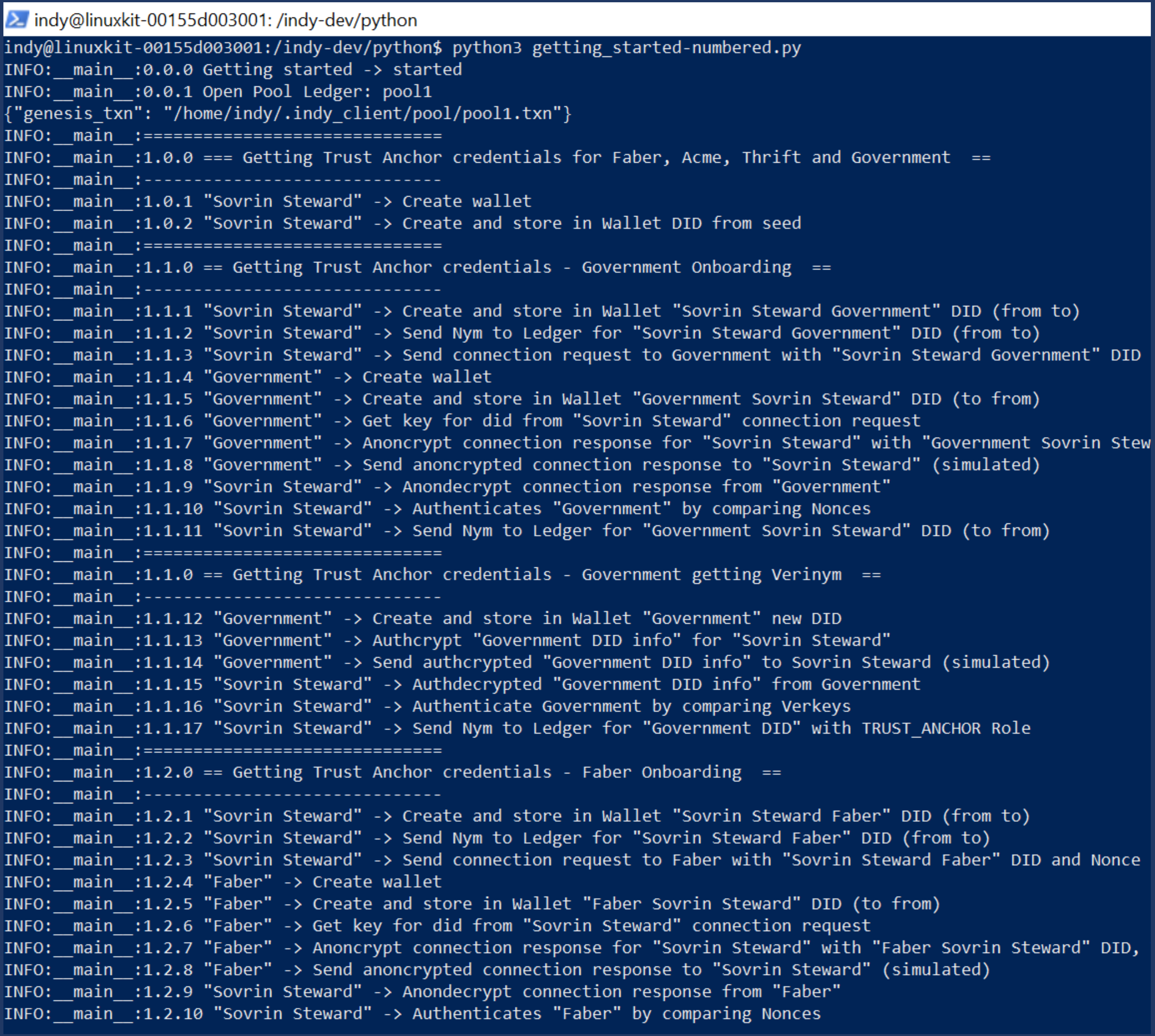
```
indy@linuxkit-00155d003001:/indy-dev/python$ cd /
indy@linuxkit-00155d003001:/$ cd indy-dev
indy@linuxkit-00155d003001:/indy-dev$ cd python
indy@linuxkit-00155d003001:/indy-dev/python$ ls
doc  getting_started-numbered.py  getting_started-verbose.py  getting_started.py  how-tos  src
indy@linuxkit-00155d003001:/indy-dev/python$ python3 getting_started-number.py
```

getting_started-numberd.py

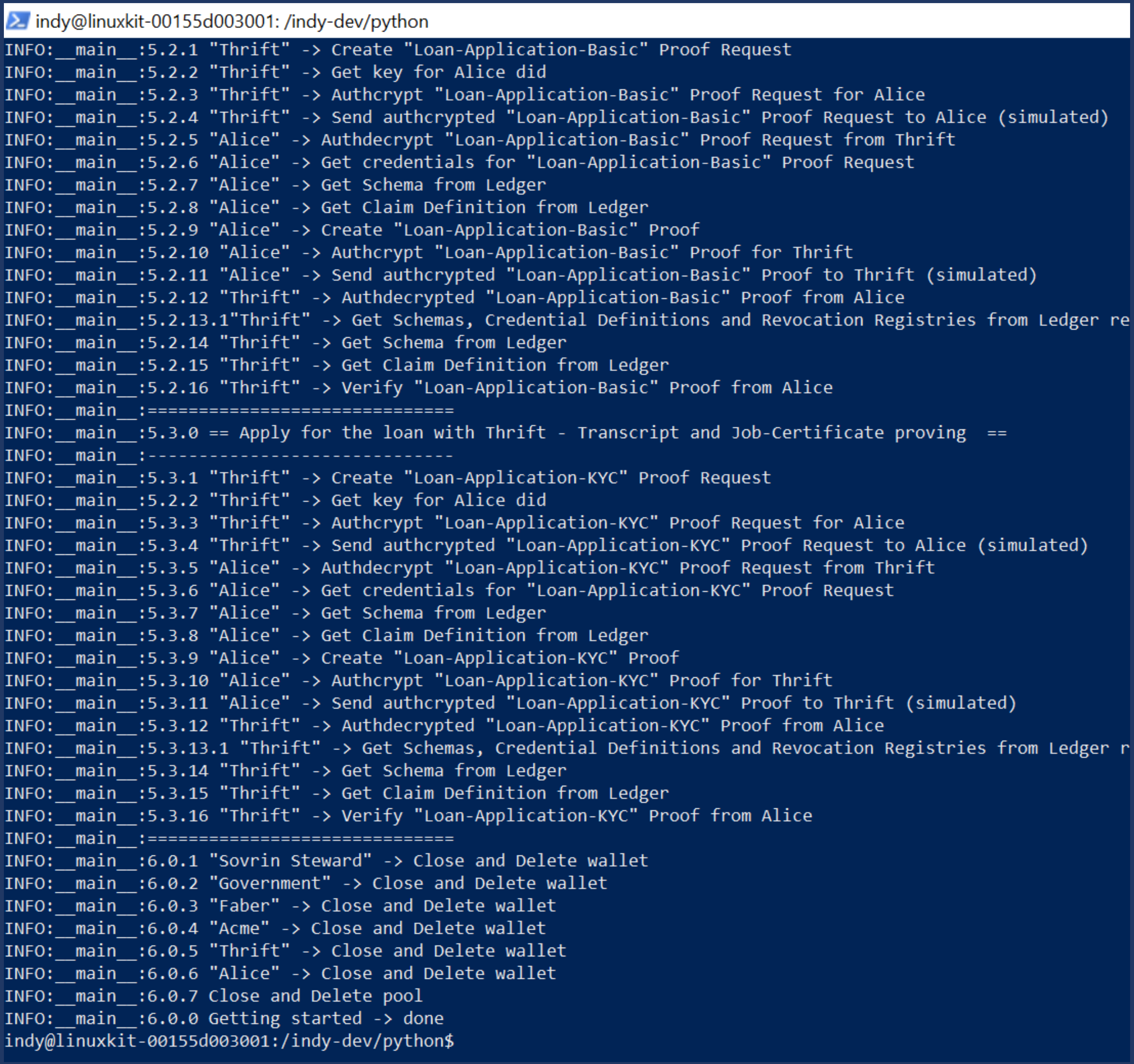


getting_started-numbered.py ×
python ▸ getting_started-numbered.py

```python
    steward_wallet = await wallet.open_wallet(steward_wallet_config, steward_wallet_credentials)

    logger.info("1.0.2 \"Sovrin Steward\" -> Create and store in Wallet DID from seed")
    steward_did_info = {'seed': '000000000000000000000000Steward1'}
    (steward_did, steward_key) = await did.create_and_store_my_did(steward_wallet, json.dumps(stewar

    logger.info("==============================")
    logger.info("1.1.0 == Getting Trust Anchor credentials - Government Onboarding  ==")
    logger.info("------------------------------")

    government_wallet_config = json.dumps({"id": "government_wallet"})
    government_wallet_credentials = json.dumps({"key": "government_wallet_key"})
    government_wallet, steward_government_key, government_steward_did, government_steward_key, _ \
        = await onboarding("1.1", pool_handle, "Sovrin Steward", steward_wallet, steward_did,
                        "Government", None, government_wallet_config, government_wallet_credentials)

    logger.info("==============================")
    logger.info("1.1.0 == Getting Trust Anchor credentials - Government getting Verinym  ==")
    logger.info("------------------------------")

    government_did = await get_verinym("1.1", pool_handle, "Sovrin Steward", steward_wallet, steward
                                       steward_government_key, "Government", government_wallet, gove
                                       government_steward_key, 'TRUST_ANCHOR')

    logger.info("==============================")
    logger.info("1.2.0 == Getting Trust Anchor credentials - Faber Onboarding  ==")
    logger.info("------------------------------")

    faber_wallet_config = json.dumps({"id": "faber_wallet"})
    faber_wallet_credentials = json.dumps({"key": "faber_wallet_key"})
```



```
indy@linuxkit-00155d003001:/indy-dev/python$ python3 getting_started-numbered.py
INFO:__main__:0.0.0 Getting started -> started
INFO:__main__:0.0.1 Open Pool Ledger: pool1
{"genesis_txn": "/home/indy/.indy_client/pool/pool1.txn"}
INFO:__main__:==============================
INFO:__main__:1.0.0 === Getting Trust Anchor credentials for Faber, Acme, Thrift and Government  ==
INFO:__main__:------------------------------
INFO:__main__:1.0.1 "Sovrin Steward" -> Create wallet
INFO:__main__:1.0.2 "Sovrin Steward" -> Create and store in Wallet DID from seed
INFO:__main__:==============================
INFO:__main__:1.1.0 == Getting Trust Anchor credentials - Government Onboarding  ==
INFO:__main__:------------------------------
INFO:__main__:1.1.1 "Sovrin Steward" -> Create and store in Wallet "Sovrin Steward Government" DID (from to)
INFO:__main__:1.1.2 "Sovrin Steward" -> Send Nym to Ledger for "Sovrin Steward Government" DID (from to)
INFO:__main__:1.1.3 "Sovrin Steward" -> Send connection request to Government with "Sovrin Steward Government" DID
INFO:__main__:1.1.4 "Government" -> Create wallet
INFO:__main__:1.1.5 "Government" -> Create and store in Wallet "Government Sovrin Steward" DID (to from)
INFO:__main__:1.1.6 "Government" -> Get key for did from "Sovrin Steward" connection request
INFO:__main__:1.1.7 "Government" -> Anoncrypt connection response for "Sovrin Steward" with "Government Sovrin Stew
INFO:__main__:1.1.8 "Government" -> Send anoncrypted connection response to "Sovrin Steward" (simulated)
INFO:__main__:1.1.9 "Sovrin Steward" -> Anondecrypt connection response from "Government"
INFO:__main__:1.1.10 "Sovrin Steward" -> Authenticates "Government" by comparing Nonces
INFO:__main__:1.1.11 "Sovrin Steward" -> Send Nym to Ledger for "Government Sovrin Steward" DID (to from)
INFO:__main__:==============================
INFO:__main__:1.1.0 == Getting Trust Anchor credentials - Government getting Verinym  ==
INFO:__main__:------------------------------
INFO:__main__:1.1.12 "Government" -> Create and store in Wallet "Government" new DID
INFO:__main__:1.1.13 "Government" -> Authcrypt "Government DID info" for "Sovrin Steward"
INFO:__main__:1.1.14 "Government" -> Send authcrypted "Government DID info" to Sovrin Steward (simulated)
INFO:__main__:1.1.15 "Sovrin Steward" -> Authdecrypted "Government DID info" from Government
INFO:__main__:1.1.16 "Sovrin Steward" -> Authenticate Government by comparing Verkeys
INFO:__main__:1.1.17 "Sovrin Steward" -> Send Nym to Ledger for "Government DID" with TRUST_ANCHOR Role
INFO:__main__:==============================
INFO:__main__:1.2.0 == Getting Trust Anchor credentials - Faber Onboarding  ==
INFO:__main__:------------------------------
INFO:__main__:1.2.1 "Sovrin Steward" -> Create and store in Wallet "Sovrin Steward Faber" DID (from to)
INFO:__main__:1.2.2 "Sovrin Steward" -> Send Nym to Ledger for "Sovrin Steward Faber" DID (from to)
INFO:__main__:1.2.3 "Sovrin Steward" -> Send connection request to Faber with "Sovrin Steward Faber" DID and Nonce
INFO:__main__:1.2.4 "Faber" -> Create wallet
INFO:__main__:1.2.5 "Faber" -> Create and store in Wallet "Faber Sovrin Steward" DID (to from)
INFO:__main__:1.2.6 "Faber" -> Get key for did from "Sovrin Steward" connection request
INFO:__main__:1.2.7 "Faber" -> Anoncrypt connection response for "Sovrin Steward" with "Faber Sovrin Steward" DID,
INFO:__main__:1.2.8 "Faber" -> Send anoncrypted connection response to "Sovrin Steward" (simulated)
INFO:__main__:1.2.9 "Sovrin Steward" -> Anondecrypt connection response from "Faber"
INFO:__main__:1.2.10 "Sovrin Steward" -> Authenticates "Faber" by comparing Nonces
```



```
INFO:__main__:5.2.1 "Thrift" -> Create "Loan-Application-Basic" Proof Request
INFO:__main__:5.2.2 "Thrift" -> Get key for Alice did
INFO:__main__:5.2.3 "Thrift" -> Authcrypt "Loan-Application-Basic" Proof Request for Alice
INFO:__main__:5.2.4 "Thrift" -> Send authcrypted "Loan-Application-Basic" Proof Request to Alice (simulated)
INFO:__main__:5.2.5 "Alice" -> Authdecrypt "Loan-Application-Basic" Proof Request from Thrift
INFO:__main__:5.2.6 "Alice" -> Get credentials for "Loan-Application-Basic" Proof Request
INFO:__main__:5.2.7 "Alice" -> Get Schema from Ledger
INFO:__main__:5.2.8 "Alice" -> Get Claim Definition from Ledger
INFO:__main__:5.2.9 "Alice" -> Create "Loan-Application-Basic" Proof
INFO:__main__:5.2.10 "Alice" -> Authcrypt "Loan-Application-Basic" Proof for Thrift
INFO:__main__:5.2.11 "Alice" -> Send authcrypted "Loan-Application-Basic" Proof to Thrift (simulated)
INFO:__main__:5.2.12 "Thrift" -> Authdecrypted "Loan-Application-Basic" Proof from Alice
INFO:__main__:5.2.13.1"Thrift" -> Get Schemas, Credential Definitions and Revocation Registries from Ledger re
INFO:__main__:5.2.14 "Thrift" -> Get Schema from Ledger
INFO:__main__:5.2.15 "Thrift" -> Get Claim Definition from Ledger
INFO:__main__:5.2.16 "Thrift" -> Verify "Loan-Application-Basic" Proof from Alice
INFO:__main__:==============================
INFO:__main__:5.3.0 == Apply for the loan with Thrift - Transcript and Job-Certificate proving  ==
INFO:__main__:------------------------------
INFO:__main__:5.3.1 "Thrift" -> Create "Loan-Application-KYC" Proof Request
INFO:__main__:5.2.2 "Thrift" -> Get key for Alice did
INFO:__main__:5.3.3 "Thrift" -> Authcrypt "Loan-Application-KYC" Proof Request for Alice
INFO:__main__:5.3.4 "Thrift" -> Send authcrypted "Loan-Application-KYC" Proof Request to Alice (simulated)
INFO:__main__:5.3.5 "Alice" -> Authdecrypt "Loan-Application-KYC" Proof Request from Thrift
INFO:__main__:5.3.6 "Alice" -> Get credentials for "Loan-Application-KYC" Proof Request
INFO:__main__:5.3.7 "Alice" -> Get Schema from Ledger
INFO:__main__:5.3.8 "Alice" -> Get Claim Definition from Ledger
INFO:__main__:5.3.9 "Alice" -> Create "Loan-Application-KYC" Proof
INFO:__main__:5.3.10 "Alice" -> Authcrypt "Loan-Application-KYC" Proof for Thrift
INFO:__main__:5.3.11 "Alice" -> Send authcrypted "Loan-Application-KYC" Proof to Thrift (simulated)
INFO:__main__:5.3.12 "Thrift" -> Authdecrypted "Loan-Application-KYC" Proof from Alice
INFO:__main__:5.3.13.1 "Thrift" -> Get Schemas, Credential Definitions and Revocation Registries from Ledger r
INFO:__main__:5.3.14 "Thrift" -> Get Schema from Ledger
INFO:__main__:5.3.15 "Thrift" -> Get Claim Definition from Ledger
INFO:__main__:5.3.16 "Thrift" -> Verify "Loan-Application-KYC" Proof from Alice
INFO:__main__:==============================
INFO:__main__:6.0.1 "Sovrin Steward" -> Close and Delete wallet
INFO:__main__:6.0.2 "Government" -> Close and Delete wallet
INFO:__main__:6.0.3 "Faber" -> Close and Delete wallet
INFO:__main__:6.0.4 "Acme" -> Close and Delete wallet
INFO:__main__:6.0.5 "Thrift" -> Close and Delete wallet
INFO:__main__:6.0.6 "Alice" -> Close and Delete wallet
INFO:__main__:6.0.7 Close and Delete pool
INFO:__main__:6.0.0 Getting started -> done
indy@linuxkit-00155d003001:/indy-dev/python$
```

Let’s look at the walkthrough on GitHub

#indygsgea

https://github.com/mwherman2000/indy-gsg-ea/blob/master/python/doc/getting_started-enterpise.md

Next Steps

- Come to Mike Lodder’s session
  - Hyperledger Aries
  - Aries Agent
  - Agent-to-Agent Communications
  - State-of-the-art Verified Credentials
  - And more

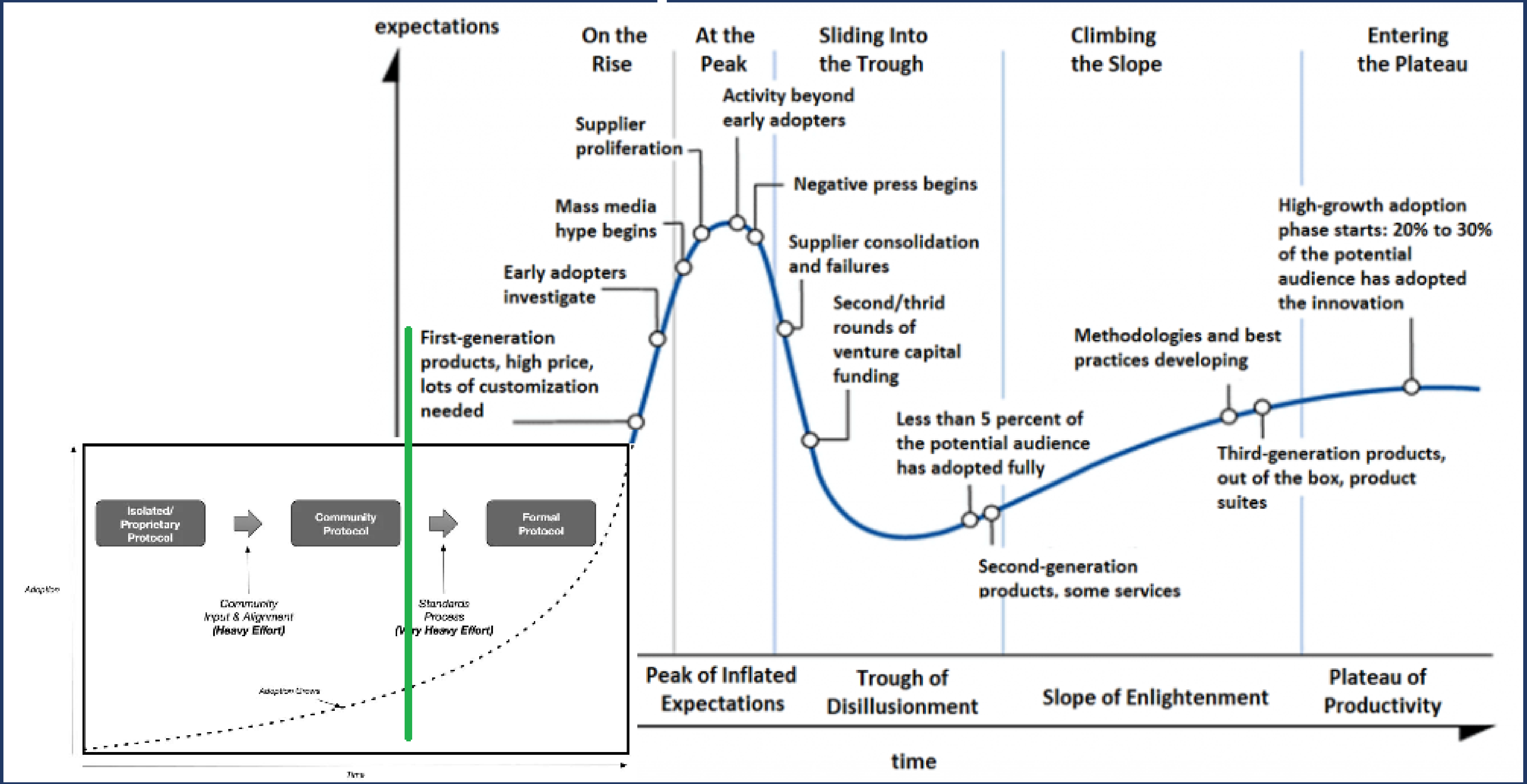
Final Note

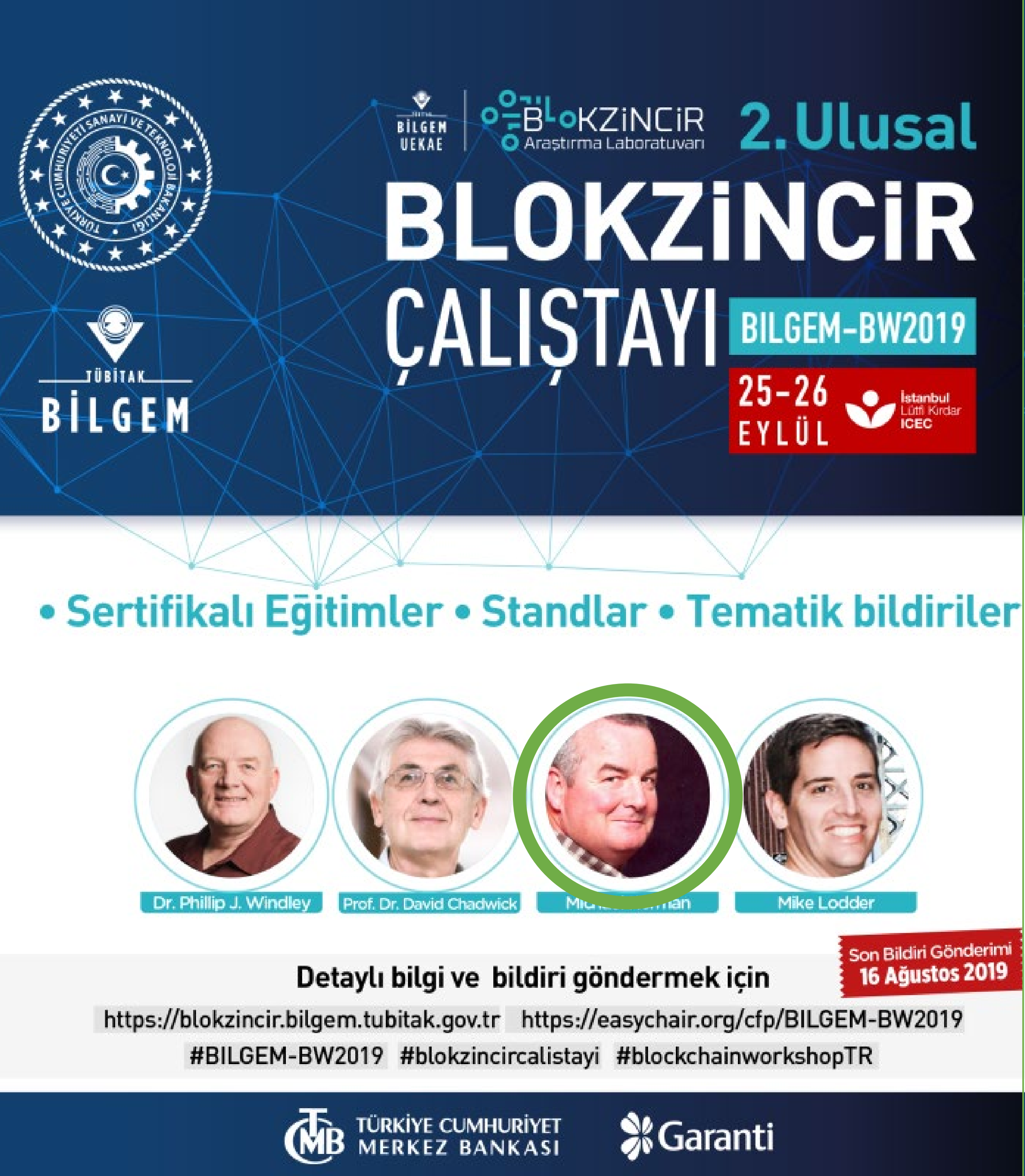
- We’re still in the very early market in relation to the use and application of blockchain technology …especially as it relates to digital identity

- While there are a number of leading specifications, implementations, and governance frameworks in the market today, I don’t believe they are close to becoming “the industry standard”. They are too new, too unproven, too risky.

- Consider other technologies that are already pervasive on the Internet, in your data centers, and in the Cloud

Final Note: Adoption


expectations
On the Rise
At the Peak
Sliding Into the Trough
Climbing the Slope
Entering the Plateau
Activity beyond early adopters
Supplier proliferation
Negative press begins
Mass media hype begins
Supplier consolidation and failures
Early adopters investigate
Second/thrid rounds of venture capital funding
First-generation products, high price, lots of customization needed
High-growth adoption phase starts: 20% to 30% of the potential audience has adopted the innovation
Methodologies and best practices developing
Less than 5 percent of the potential audience has adopted fully
Third-generation products, out of the box, product suites
Second-generation products, some services
Peak of Inflated Expectations
Trough of Disillusionment
Slope of Enlightenment
Plateau of Productivity
time
Isolated/ Proprietary Protocol
Community Protocol
Formal Protocol
Community Input & Alignment (Heavy Effort)
Standards Process (Very Heavy Effort)
Adoption
Adoption Grows
Time



BİLGEM UEKAE

BLOKZİNCİR Araştırma Laboratuvarı

2. Ulusal

BLOKZİNCİR ÇALIŞTAYI

BILGEM-BW2019

25-26 EYLÜL

İstanbul Lütfi Kırdar ICEC

TÜBİTAK BİLGEM

• Sertifikalı Eğitimler • Standlar • Tematik bildiriler

Dr. Phillip J. Windley

Prof. Dr. David Chadwick

Mi[illegible]man

Mike Lodder

Son Bildiri Gönderimi 16 Ağustos 2019

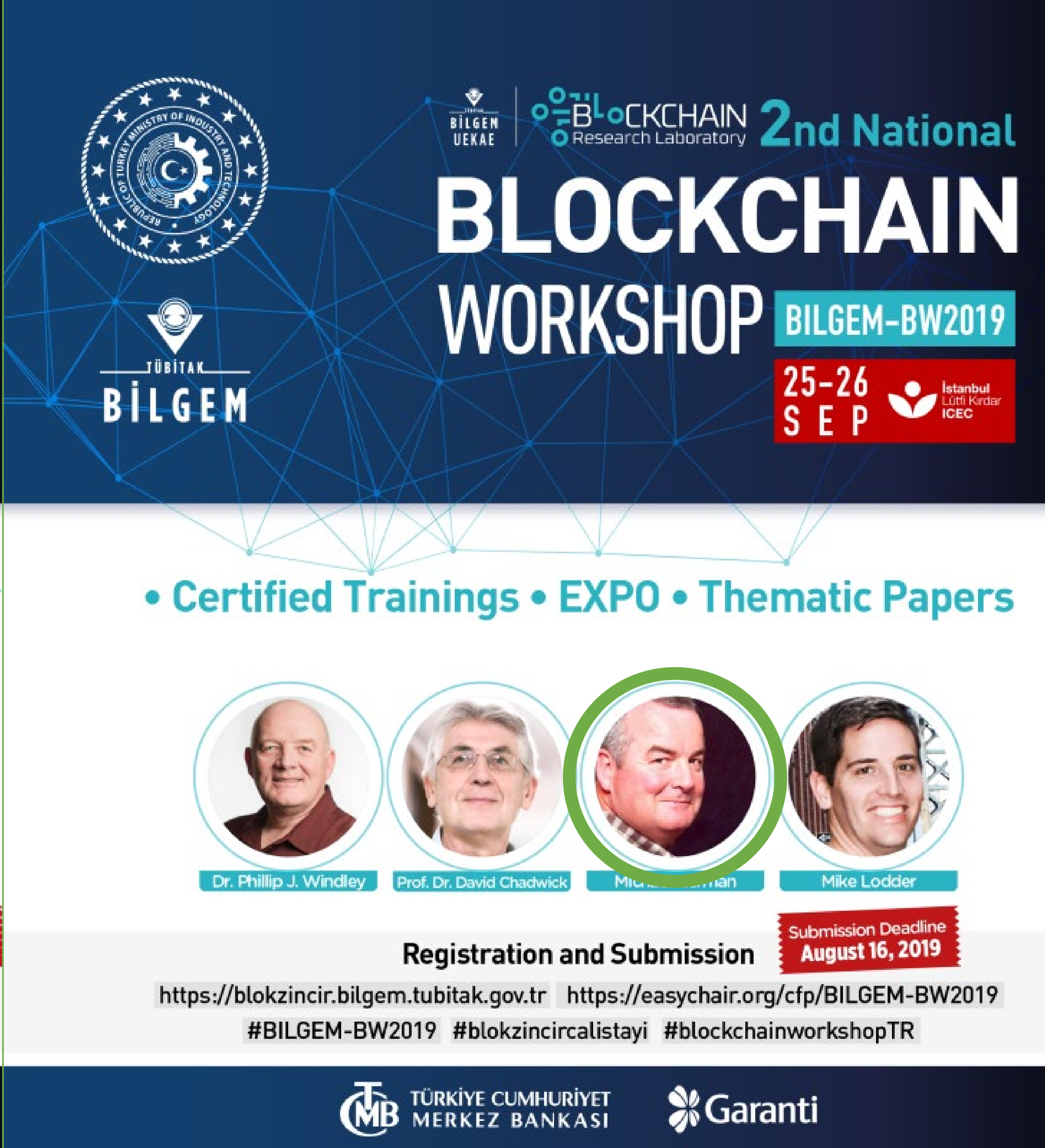
Detaylı bilgi ve bildiri göndermek için

https://blokzincir.bilgem.tubitak.gov.tr https://easychair.org/cfp/BILGEM-BW2019

#BILGEM-BW2019 #blokzincircalistayi #blockchainworkshopTR

TÜRKİYE CUMHURİYET MERKEZ BANKASI

Garanti



BİLGEM UEKAE

BLOCKCHAIN Research Laboratory

2nd National

BLOCKCHAIN WORKSHOP

BILGEM-BW2019

25-26 SEP

İstanbul Lütfi Kırdar ICEC

TÜBİTAK BİLGEM

• Certified Trainings • EXPO • Thematic Papers

Dr. Phillip J. Windley

Prof. Dr. David Chadwick

Mi[illegible]man

Mike Lodder

Submission Deadline August 16, 2019

Registration and Submission

https://blokzincir.bilgem.tubitak.gov.tr https://easychair.org/cfp/BILGEM-BW2019

#BILGEM-BW2019 #blokzincircalistayi #blockchainworkshopTR

TÜRKİYE CUMHURİYET MERKEZ BANKASI

Garanti

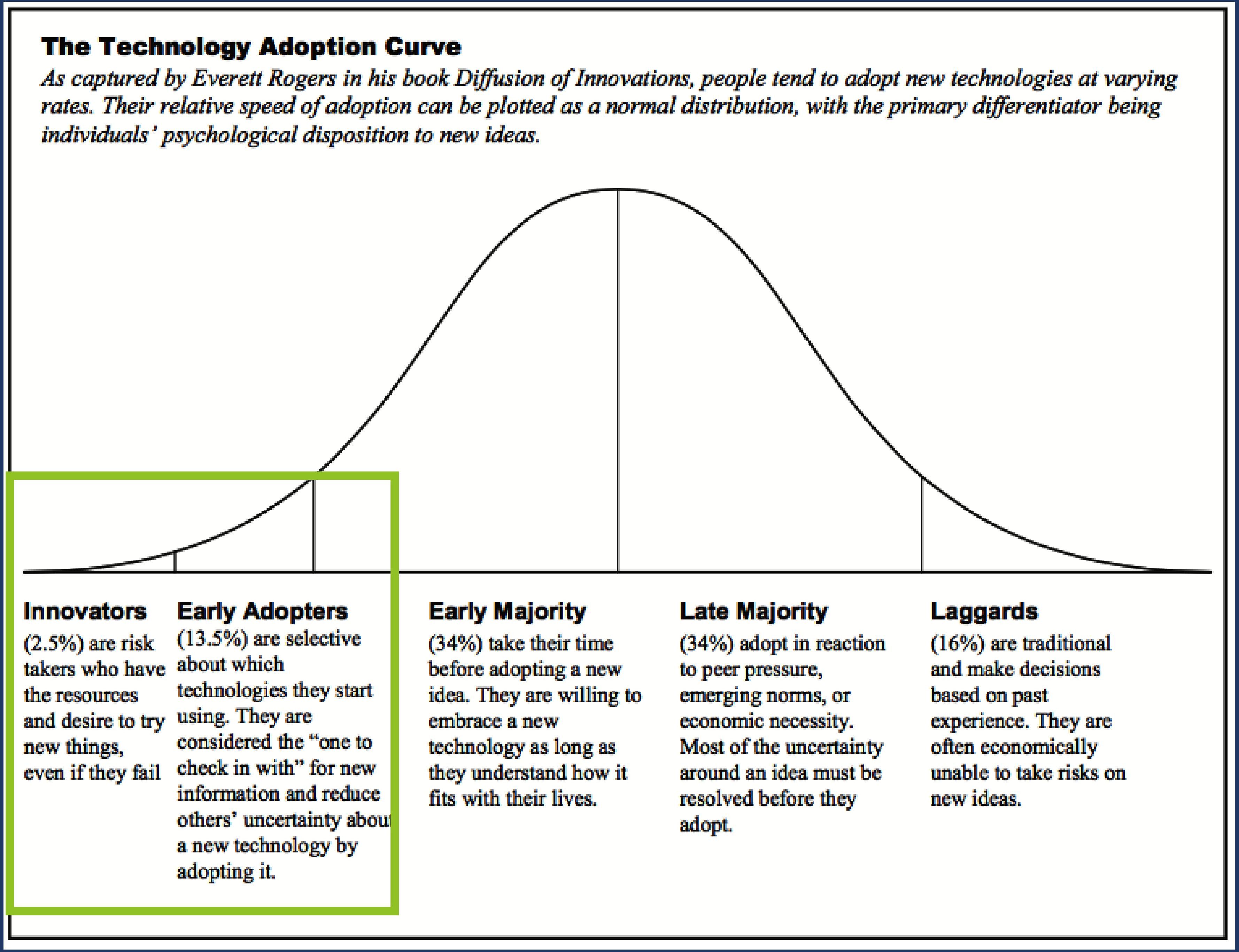
Questions?

mwherman@parallelspace.net

Backup Slides

Final Note

Early Adopters Marketplace



The Technology Adoption Curve

As captured by Everett Rogers in his book Diffusion of Innovations, people tend to adopt new technologies at varying rates. Their relative speed of adoption can be plotted as a normal distribution, with the primary differentiator being individuals' psychological disposition to new ideas.

Innovators
(2.5%) are risk takers who have the resources and desire to try new things, even if they fail

Early Adopters
(13.5%) are selective about which technologies they start using. They are considered the "one to check in with" for new information and reduce others' uncertainty about a new technology by adopting it.

Early Majority
(34%) take their time before adopting a new idea. They are willing to embrace a new technology as long as they understand how it fits with their lives.

Late Majority
(34%) adopt in reaction to peer pressure, emerging norms, or economic necessity. Most of the uncertainty around an idea must be resolved before they adopt.

Laggards
(16%) are traditional and make decisions based on past experience. They are often economically unable to take risks on new ideas.